.



Facultad de Ingeniería

Diseño y Arquitectura de Software

Proyecto final

Estudiantes:

Julian Ballesteros: 0000154065

Santiago Cepeda: 0000151059

Juan Camilo Posada: 0000153647

Docente: Juan Pablo Garzon Ruiz

Jueves 21 de Noviembre del 2018

1. ***Tablas de casos de uso unitarios (individual, dependiendo de la funcionalidad asignada)***

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | testValorPago1 | | |
| Identificador caso de prueba | CP0001\_testValorPago1 | | |
| Función probar | Asignar el valor de un pago a realizar | | |
| Objetivo | Verificar la correcta asignación del valor a pagar | | |
| Descripción | Se verifica que el precio del pago quede correctamente asignado | | |
| Criterios de éxito | El valor actual es el mismo al registrado dentro del sistema | | |
| Criterios de falla | El valor actual difiere del valor guardado | | |
| Precondiciones | Tener todos los atributos del pago a ingresar | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Tener un pago a realizar, objeto tipo pago | | |
| Autor | Juan Posada | | |
| Fecha de creación | 21-11-2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Ingresar los valores del pago | - |
| 2 | Realizar pago | Pago realizado exitosamente |
|  |  |  |
| Post condiciones | El pago queda registrado en el sistema | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | testValorPago2 | | |
| Identificador caso de prueba | CP0002\_testValorPago2 | | |
| Función probar | Asignar el valor de un pago a realizar | | |
| Objetivo | Verificar la correcta asignación del valor a pagar | | |
| Descripción | Se verifica que el precio del pago quede correctamente asignado | | |
| Criterios de éxito | El valor actual es el mismo al registrado dentro del sistema | | |
| Criterios de falla | El valor actual difiere del valor guardado | | |
| Precondiciones | Tener todos los atributos del pago a ingresar | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Tener un pago a realizar, objeto tipo pago | | |
| Autor | Juan Posada | | |
| Fecha de creación | 21-11-2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Ingresar los valores del pago | - |
| 2 | Realizar pago | Pago realizado exitosamente |
|  |  |  |
| Post condiciones | El pago queda registrado en el sistema | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | testValorPago3 | | |
| Identificador caso de prueba | CP0003\_testValorPago3 | | |
| Función probar | Asignar el valor de un pago a realizar | | |
| Objetivo | Verificar la correcta asignación del valor a pagar | | |
| Descripción | Se verifica que el precio del pago quede correctamente asignado | | |
| Criterios de éxito | El valor actual es el mismo al registrado dentro del sistema | | |
| Criterios de falla | El valor actual difiere del valor guardado | | |
| Precondiciones | Tener todos los atributos del pago a ingresar | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Tener un pago a realizar, objeto tipo pago | | |
| Autor | Juan Posada | | |
| Fecha de creación | 21-11-2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Ingresar los valores del pago | - |
| 2 | Realizar pago | Pago realizado exitosamente |
|  |  |  |
| Post condiciones | El pago queda registrado en el sistema | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | testRefPago1 | | |
| Identificador caso de prueba | CP0001\_testRefPago1 | | |
| Función probar | Asignar el número de referencia de un pago | | |
| Objetivo | Verificar la correcta asignación de referencia | | |
| Descripción | Se verifica que la referencia quede registrada a la transacción de pago correcta | | |
| Criterios de éxito | El número de referencia actual es el mismo al que se encuentra almacenado en el sistema | | |
| Criterios de falla | El número de referencia actual difiere del que se encuentra almacenado | | |
| Precondiciones | Tener todos los atributos del pago a ingresar | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Tener un pago a realizar, objeto tipo pago | | |
| Autor | Juan Posada | | |
| Fecha de creación | 21-11-2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Ingresar los valores del pago | - |
| 2 | Realizar pago | Pago realizado exitosamente |
|  |  |  |
| Post condiciones | El pago queda registrado en el sistema | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | testRefPago2 | | |
| Identificador caso de prueba | CP0002\_testRefPago2 | | |
| Función probar | Asignar el número de referencia de un pago | | |
| Objetivo | Verificar la correcta asignación de referencia | | |
| Descripción | Se verifica que la referencia quede registrada a la transacción de pago correcta | | |
| Criterios de éxito | El número de referencia actual es el mismo al que se encuentra almacenado en el sistema | | |
| Criterios de falla | El número de referencia actual difiere del que se encuentra almacenado | | |
| Precondiciones | Tener todos los atributos del pago a ingresar | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Tener un pago a realizar, objeto tipo pago | | |
| Autor | Juan Posada | | |
| Fecha de creación | 21-11-2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Ingresar los valores del pago | - |
| 2 | Realizar pago | Pago realizado exitosamente |
|  |  |  |
| Post condiciones | El pago queda registrado en el sistema | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | testRefPago3 | | |
| Identificador caso de prueba | CP0003\_testRefPago3 | | |
| Función probar | Asignar el número de referencia de un pago | | |
| Objetivo | Verificar la correcta asignación de referencia | | |
| Descripción | Se verifica que la referencia quede registrada a la transacción de pago correcta | | |
| Criterios de éxito | El número de referencia actual es el mismo al que se encuentra almacenado en el sistema | | |
| Criterios de falla | El número de referencia actual difiere del que se encuentra almacenado | | |
| Precondiciones | Tener todos los atributos del pago a ingresar | | |
| Perfil del usuario | El número de referencia actual difiere del que se encuentra almacenado | | |
| Necesidades para el caso de prueba | Tener un pago a realizar, objeto tipo pago | | |
| Autor | Juan Posada | | |
| Fecha de creación | 21-11-2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Ingresar los valores del pago | - |
| 2 | Realizar pago | Pago realizado exitosamente |
|  |  |  |
| Post condiciones | El pago queda registrado en el sistema | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | testUsuarioPaga1 | | |
| Identificador caso de prueba | CP0001\_testUsuarioPaga1 | | |
| Función probar | Asignar el usuario que realiza el pago | | |
| Objetivo | Verificar la correcta asignación de usuario que realiza el pago | | |
| Descripción | Asignar el id de usuario que realiza el pago de la transacción | | |
| Criterios de éxito | El id actual es el mismo al que está almacenado | | |
| Criterios de falla | El id actual difiere del almacenado | | |
| Precondiciones | Tener todos los atributos del pago a ingresar | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Tener un pago a realizar, objeto tipo pago | | |
| Autor | Juan Posada | | |
| Fecha de creación | 21-11-2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Ingresar los valores del pago | - |
| 2 | Realizar pago | Pago realizado exitosamente |
|  |  |  |
| Post condiciones | El pago queda registrado en el sistema | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | testUsuarioPaga2 | | |
| Identificador caso de prueba | CP0002\_testUsuarioPaga2 | | |
| Función probar | Asignar el usuario que realiza el pago | | |
| Objetivo | Verificar la correcta asignación de usuario que realiza el pago | | |
| Descripción | Asignar el id de usuario que realiza el pago de la transacción | | |
| Criterios de éxito | El id actual es el mismo al que está almacenado | | |
| Criterios de falla | El id actual difiere del almacenado | | |
| Precondiciones | Tener todos los atributos del pago a ingresar | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Tener un pago a realizar, objeto tipo pago | | |
| Autor | Juan Posada | | |
| Fecha de creación | 21-11-2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Ingresar los valores del pago | - |
| 2 | Realizar pago | Pago realizado exitosamente |
|  |  |  |
| Post condiciones | El pago queda registrado en el sistema | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | testUsuarioPaga3 | | |
| Identificador caso de prueba | CP0003\_testUsuarioPaga | | |
| Función probar | Asignar el usuario que realiza el pago | | |
| Objetivo | Verificar la correcta asignación de usuario que realiza el pago | | |
| Descripción | Asignar el id de usuario que realiza el pago de la transacción | | |
| Criterios de éxito | El id actual es el mismo al que está almacenado | | |
| Criterios de falla | El id actual difiere del almacenado | | |
| Precondiciones | Tener todos los atributos del pago a ingresar | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Tener un pago a realizar, objeto tipo pago | | |
| Autor | Juan Posada | | |
| Fecha de creación | 21-11-2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Ingresar los valores del pago | - |
| 2 | Realizar pago | Pago realizado exitosamente |
|  |  |  |
| Post condiciones | El pago queda registrado en el sistema | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | testUsuarioPagado1 | | |
| Identificador caso de prueba | CP0001\_testUsuarioPagado1 | | |
| Función probar | Asignar el usuario que recibe el pago | | |
| Objetivo | Verificar la correcta asignación de usuario que recibe el pago | | |
| Descripción | Asignar el id de usuario que recibe el pago de la transacción | | |
| Criterios de éxito | El id actual es el mismo al que está almacenado | | |
| Criterios de falla | El id actual difiere del almacenado | | |
| Precondiciones | Tener todos los atributos del pago a ingresar | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Tener un pago a realizar, objeto tipo pago | | |
| Autor | Juan Posada | | |
| Fecha de creación | 21-11-2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Ingresar los valores del pago | - |
| 2 | Realizar pago | Pago realizado exitosamente |
|  |  |  |
| Post condiciones | El pago queda registrado en el sistema | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | testUsuarioPagado2 | | |
| Identificador caso de prueba | CP0002\_testUsuarioPagado2 | | |
| Función probar | Asignar el usuario que recibe el pago | | |
| Objetivo | Verificar la correcta asignación de usuario que recibe el pago | | |
| Descripción | Asignar el id de usuario que recibe el pago de la transacción | | |
| Criterios de éxito | El id actual es el mismo al que está almacenado | | |
| Criterios de falla | El id actual difiere del almacenado | | |
| Precondiciones | Tener todos los atributos del pago a ingresar | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Tener un pago a realizar, objeto tipo pago | | |
| Autor | Juan Posada | | |
| Fecha de creación | 21-11-2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Ingresar los valores del pago | - |
| 2 | Realizar pago | Pago realizado exitosamente |
|  |  |  |
| Post condiciones | El pago queda registrado en el sistema | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | testUsuarioPagado3 | | |
| Identificador caso de prueba | CP0003\_testUsuarioPagado3 | | |
| Función probar | Asignar el usuario que recibe el pago | | |
| Objetivo | Verificar la correcta asignación de usuario que recibe el pago | | |
| Descripción | Asignar el id de usuario que recibe el pago de la transacción | | |
| Criterios de éxito | El id actual es el mismo al que está almacenado | | |
| Criterios de falla | El id actual difiere del almacenado | | |
| Precondiciones | Tener todos los atributos del pago a ingresar | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Tener un pago a realizar, objeto tipo pago | | |
| Autor | Juan Posada | | |
| Fecha de creación | 21-11-2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Ingresar los valores del pago | - |
| 2 | Realizar pago | Pago realizado exitosamente |
|  |  |  |
| Post condiciones | El pago queda registrado en el sistema | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | testConceptoPago1 | | |
| Identificador caso de prueba | CP0001\_testConceptoPago1 | | |
| Función probar | Asignar el concepto del pago | | |
| Objetivo | Verificar la correcta asignación del concepto de pago | | |
| Descripción | Asignar el concepto del pago a realizar | | |
| Criterios de éxito | El concepto actual coincide con el concepto almacenado | | |
| Criterios de falla | El concepto actual difiere del almacenado | | |
| Precondiciones | Tener todos los atributos del pago a ingresar | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Tener un pago a realizar, objeto tipo pago | | |
| Autor | Juan Posada | | |
| Fecha de creación | 21-11-2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Ingresar los valores del pago | - |
| 2 | Realizar pago | Pago realizado exitosamente |
|  |  |  |
| Post condiciones | El pago queda registrado en el sistema | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | testConceptoPago2 | | |
| Identificador caso de prueba | CP0002\_testConceptoPago2 | | |
| Función probar | Asignar el concepto del pago | | |
| Objetivo | Verificar la correcta asignación del concepto de pago | | |
| Descripción | Asignar el concepto del pago a realizar | | |
| Criterios de éxito | El concepto actual coincide con el concepto almacenado | | |
| Criterios de falla | El concepto actual difiere del almacenado | | |
| Precondiciones | Tener todos los atributos del pago a ingresar | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Tener un pago a realizar, objeto tipo pago | | |
| Autor | Juan Posada | | |
| Fecha de creación | 21-11-2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Ingresar los valores del pago | - |
| 2 | Realizar pago | Pago realizado exitosamente |
|  |  |  |
| Post condiciones | El pago queda registrado en el sistema | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | testConceptoPago3 | | |
| Identificador caso de prueba | CP0003\_testConceptoPago3 | | |
| Función probar | Asignar el concepto del pago | | |
| Objetivo | Verificar la correcta asignación del concepto de pago | | |
| Descripción | Asignar el concepto del pago a realizar | | |
| Criterios de éxito | El concepto actual coincide con el concepto almacenado | | |
| Criterios de falla | El concepto actual difiere del almacenado | | |
| Precondiciones | Tener todos los atributos del pago a ingresar | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Tener un pago a realizar, objeto tipo pago | | |
| Autor | Juan Posada | | |
| Fecha de creación | 21-11-2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Ingresar los valores del pago | - |
| 2 | Realizar pago | Pago realizado exitosamente |
|  |  |  |
| Post condiciones | El pago queda registrado en el sistema | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | crearPasajeroTest | | |
| Identificador caso de prueba | CP001\_CrearPasajero | | |
| Función probar | Agregar pasajeros a el arreglo de el proxy | | |
| Objetivo | Verificar que el Pasajero se agregue correctamente a el arreglo de usuarios | | |
| Descripción | Probar si el Pasajero se añade a el arreglo con los atributos ingresados | | |
| Criterios de éxito | El pasajero queda agregado en el arreglo de usuarios | | |
| Criterios de falla | El pasajero no se agregó a el arreglo de usuarios | | |
| Precondiciones | Tener los datos de el pasajero a agregar | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Ingresar todos los datos requeridos para el usuario(Correo, contraseña, nombre, apellido, edad,id) | | |
| Autor | Julian Ballesteros | | |
| Fecha de creación | 21 de Noviembre del 2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Asignar los datos de el pasajero | - |
| 2 | Crear el pasajero | Pasajero creado y se agregó a el arreglo |
|  |  |  |
| Post condiciones | El Pasajero quedará registrado en la plataforma y podrá iniciar sesión | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | crearConductorTest | | |
| Identificador caso de prueba | CP001\_CrearConductor | | |
| Función probar | Agregar pasajeros a el arreglo de el proxy | | |
| Objetivo | Verificar que el Conductor se agregue correctamente a el arreglo de usuarios | | |
| Descripción | Probar si el Conductor se añade a el arreglo con los atributos ingresados | | |
| Criterios de éxito | El conductor queda agregado en el arreglo de usuarios | | |
| Criterios de falla | El conductor no se agregó a el arreglo de usuarios | | |
| Precondiciones | Tener los datos de el conductor a agregar | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Ingresar todos los datos requeridos para el usuario(Correo, contraseña, nombre, apellido, edad,id) | | |
| Autor | Julian Ballesteros | | |
| Fecha de creación | 21 de Noviembre del 2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Asignar los datos de el conductor | - |
| 2 | Crear el conductor | Conductor creado y se agregó a el arreglo |
|  |  |  |
| Post condiciones | El Pasajero quedará registrado en la plataforma y podrá iniciar sesión | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | authTest1 | | |
| Identificador caso de prueba | CP001\_AuthTest1 | | |
| Función probar | Inicio sesión | | |
| Objetivo | Verificar que usuarios que ingresen credenciales incorrectos no puedan acceder a la plataforma | | |
| Descripción | Probar si el sistema valida el usuario de manera correcta y si no lo encuentra no retorna número de sesión por lo cual no tiene acceso a el sistema. | | |
| Criterios de éxito | El sistema retorna un mensaje de error diciendo que no se pudo encontrar el usuario | | |
| Criterios de falla | El sistema retorna número de sesión indicando que ese usuario tiene acceso a el sistema | | |
| Precondiciones | Usuario NO ha creado y registrado en el arreglo de usuarios | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | El usuario NO existe en el sistema | | |
| Autor | Julian Ballesteros | | |
| Fecha de creación | 21 de noviembre del 2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Ingresa credenciales de incorrectos que no existen en el sistema | Retorna el mensaje de error ya que no pudo encontrar el usuario |
|  |  |  |
|  |  |  |
| Post condiciones | El usuario ya tiene el numero de sesion con el cual podrá acceder a el sistema a realizar los pagos y consultarlos. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | authTest2 | | |
| Identificador caso de prueba | CP001\_AuthTest1 | | |
| Función probar | Inicio sesión | | |
| Objetivo | Verificar que el usuario pueda iniciar sesión a la plataforma con sus credenciales | | |
| Descripción | Probar si el sistema valida el usuario de manera correcta y retorna el numero de sesion | | |
| Criterios de éxito | El sistema retorna el número de sesión indicando que los credenciales son correctos. | | |
| Criterios de falla | El sistema lanza una excepción explicando que no se encontró el usuario. | | |
| Precondiciones | Usuario creado y registrado en el arreglo de usuarios | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | El usuario existe en el sistema | | |
| Autor | Julian Ballesteros | | |
| Fecha de creación | 21 de noviembre del 2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Ingresa credenciales de pasajero correctos que ya existen en el sistema | Retorna el número de sesión asignado a este usuario |
|  |  |  |
|  |  |  |
| Post condiciones | El usuario ya tiene el numero de sesion con el cual podrá acceder a el sistema a realizar los pagos y consultarlos. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | authTest3 | | |
| Identificador caso de prueba | CP001\_AuthTest3 | | |
| Función probar | Inicio sesión | | |
| Objetivo | Verificar que el usuario pueda iniciar sesión a la plataforma con sus credenciales | | |
| Descripción | Probar si el sistema valida el usuario de manera correcta y retorna el numero de sesion | | |
| Criterios de éxito | El sistema retorna el número de sesión indicando que los credenciales son correctos. | | |
| Criterios de falla | El sistema lanza una excepción explicando que no se encontró el usuario. | | |
| Precondiciones | Usuario creado y registrado en el arreglo de usuarios | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | El usuario existe en el sistema | | |
| Autor | Julian Ballesteros | | |
| Fecha de creación | 21 de noviembre del 2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Ingresa credenciales de conductor correctos que ya existen en el sistema | Retorna el número de sesión asignado a este usuario |
|  |  |  |
|  |  |  |
| Post condiciones | El usuario ya tiene el numero de sesion con el cual podrá acceder a el sistema a realizar los pagos y consultarlos. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | guardarSesionesTest | | |
| Identificador caso de prueba | CP001\_GuardarSesionesTest | | |
| Función probar | Las sesiones de usuarios que ingresaron a el sistema con credenciales correctos se almacenan en el arreglo de sesiones | | |
| Objetivo | Verificar que los usuarios que tienen acceso al sistema una vez inician sesión se guarda el numero de sesion para que puedan realizar las demás operaciones | | |
| Descripción | Probar si se almacena la sesión en el arreglo | | |
| Criterios de éxito | La sesión se almacena y retorna de manera correcta en el sistema | | |
| Criterios de falla | La sesión no se guarda en el arreglo y no se retorna. | | |
| Precondiciones | - | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Tener numero de sesion a registrar en el arreglo | | |
| Autor | Santiago Cepeda | | |
| Fecha de creación | 21 de noviembre del 2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | asignarle el numero de sesion a el objeto tipo Session | - |
| 2 | ejecutar el método de guardado de sesión. | Retorna el número de sesión almacenado en el arreglo |
|  |  |  |
| Post condiciones | la sesión queda almacenada en el sistema | | |

1. ***Tablas de casos de uso integración y sistema (entre los integrantes del grupo y las de sistema entre todos).***

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | testFechaPago1 | | |
| Identificador caso de prueba | CP0001\_testFechaPago1 | | |
| Función probar | Asignación de fecha al momento de realizar un pago en efectivo | | |
| Objetivo | Asignar la fecha en la que un pago en efectivo fue realizado | | |
| Descripción | Todo pago al realizarse, genera automáticamente la fecha en la que fue realizado | | |
| Criterios de éxito | La fecha de pago actual coincide con la que se encuentra almacenada | | |
| Criterios de falla | La fecha de pago difiere con la que se encuentra almacenada | | |
| Precondiciones | Tener un pago, para realizar su respectiva transacción | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Objeto tipo pago, tipo Efectivo | | |
| Autor | Juan Posada | | |
| Fecha de creación | 21-11-2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Se genera un pago a realizar | Asigna los atributos del pago |
| 2 | Se realiza el pago en efectivo | Transacción realizada |
| 3 | Se le asigna su fecha al pago | Generación automática de la fecha en que se realizó la transacción |
| 4 | Finaliza el pago | Pago realizado exitosamente |
| Post condiciones | El sistema registra que el pago fue realizado, en la fecha correspondiente. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | testFechaPago2 | | |
| Identificador caso de prueba | CP0002\_testFechaPago2 | | |
| Función probar | Asignación de fecha al momento de realizar un pago en efectivo | | |
| Objetivo | Asignar la fecha en la que un pago en efectivo fue realizado | | |
| Descripción | Todo pago al realizarse, genera automáticamente la fecha en la que fue realizado | | |
| Criterios de éxito | La fecha de pago actual coincide con la que se encuentra almacenada | | |
| Criterios de falla | La fecha de pago difiere con la que se encuentra almacenada | | |
| Precondiciones | Tener un pago, para realizar su respectiva transacción | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Objeto tipo pago, tipo Efectivo | | |
| Autor | Juan Posada | | |
| Fecha de creación | 21-11-2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Se genera un pago a realizar | Asigna los atributos del pago |
| 2 | Se realiza el pago en efectivo | Transacción realizada |
| 3 | Se le asigna su fecha al pago | Generación automática de la fecha en que se realizó la transacción |
| 4 | Finaliza el pago | Pago realizado exitosamente |
| Post condiciones | El sistema registra que el pago fue realizado, en la fecha correspondiente. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | testFechaPago3 (Efectivo) | | |
| Identificador caso de prueba | CP0003\_testFechaPago3 | | |
| Función probar | Asignación de fecha al momento de realizar un pago en efectivo | | |
| Objetivo | Asignar la fecha en la que un pago en efectivo fue realizado | | |
| Descripción | Todo pago al realizarse, genera automáticamente la fecha en la que fue realizado | | |
| Criterios de éxito | La fecha de pago actual coincide con la que se encuentra almacenada | | |
| Criterios de falla | La fecha de pago difiere con la que se encuentra almacenada | | |
| Precondiciones | Tener un pago, para realizar su respectiva transacción | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Objeto tipo pago, tipo Efectivo | | |
| Autor | Juan Posada | | |
| Fecha de creación | 21-11-2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Se genera un pago a realizar | Asigna los atributos del pago |
| 2 | Se realiza el pago en efectivo | Transacción realizada |
| 3 | Se le asigna su fecha al pago | Generación automática de la fecha en que se realizó la transacción |
| 4 | Finaliza el pago | Pago realizado exitosamente |
| Post condiciones | El sistema registra que el pago fue realizado, en la fecha correspondiente. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | testFechaPago4 | | |
| Identificador caso de prueba | CP0004\_testFechaPago4 | | |
| Función probar | Asignación de fecha al momento de realizar un pago por crédito | | |
| Objetivo | Asignar la fecha en la que un pago por crédito fue realizado | | |
| Descripción | Todo pago al realizarse, genera automáticamente la fecha en la que fue realizado | | |
| Criterios de éxito | La fecha de pago actual coincide con la que se encuentra almacenada | | |
| Criterios de falla | La fecha de pago difiere con la que se encuentra almacenada | | |
| Precondiciones | Tener un pago, para realizar su respectiva transacción | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Objeto tipo pago, tipo Crédito | | |
| Autor | Juan Posada | | |
| Fecha de creación | 21-11-2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Se genera un pago a realizar | Asigna los atributos del pago |
| 2 | Se realiza el pago por crédito | Transacción realizada |
| 3 | Se le asigna su fecha al pago | Generación automática de la fecha en que se realizó la transacción |
| 4 | Finaliza el pago | Pago realizado exitosamente |
| Post condiciones | El sistema registra que el pago fue realizado, en la fecha correspondiente. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | testFechaPago5 | | |
| Identificador caso de prueba | CP0005\_testFechaPago5 | | |
| Función probar | Asignación de fecha al momento de realizar un pago por crédito | | |
| Objetivo | Asignar la fecha en la que un pago por crédito fue realizado | | |
| Descripción | Todo pago al realizarse, genera automáticamente la fecha en la que fue realizado | | |
| Criterios de éxito | La fecha de pago actual coincide con la que se encuentra almacenada | | |
| Criterios de falla | La fecha de pago difiere con la que se encuentra almacenada | | |
| Precondiciones | Tener un pago, para realizar su respectiva transacción | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Objeto tipo pago, tipo Crédito | | |
| Autor | Juan Posada | | |
| Fecha de creación | 21-11-2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Se genera un pago a realizar | Asigna los atributos del pago |
| 2 | Se realiza el pago por crédito | Transacción realizada |
| 3 | Se le asigna su fecha al pago | Generación automática de la fecha en que se realizó la transacción |
| 4 | Finaliza el pago | Pago realizado exitosamente |
| Post condiciones | El sistema registra que el pago fue realizado, en la fecha correspondiente. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | testFechaPago6 | | |
| Identificador caso de prueba | CP0006\_testFechaPago6 | | |
| Función probar | Asignación de fecha al momento de realizar un pago por crédito | | |
| Objetivo | Asignar la fecha en la que un pago por crédito fue realizado | | |
| Descripción | Todo pago al realizarse, genera automáticamente la fecha en la que fue realizado | | |
| Criterios de éxito | La fecha de pago actual coincide con la que se encuentra almacenada | | |
| Criterios de falla | La fecha de pago difiere con la que se encuentra almacenada | | |
| Precondiciones | Tener un pago, para realizar su respectiva transacción | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Objeto tipo pago, tipo Crédito | | |
| Autor | Juan Posada | | |
| Fecha de creación | 21-11-2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Se genera un pago a realizar | Asigna los atributos del pago |
| 2 | Se realiza el pago por crédito | Transacción realizada |
| 3 | Se le asigna su fecha al pago | Generación automática de la fecha en que se realizó la transacción |
| 4 | Finaliza el pago | Pago realizado exitosamente |
| Post condiciones | El sistema registra que el pago fue realizado, en la fecha correspondiente. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | testTarjetaPago1 | | |
| Identificador caso de prueba | CP0001\_testTarjetaPago1 | | |
| Función probar | Asignación de números de tarjeta de crédito a los pagos | | |
| Objetivo | Verificar que el número de tarjeta de crédito ingresado sea el correcto | | |
| Descripción | Al momento de realizarse un pago con tarjeta de crédito, uno de los atributos más importantes del pago por este método es este. | | |
| Criterios de éxito | El número de tarjeta ingresado coincide con el almacenado | | |
| Criterios de falla | El número de tarjeta difiere con el almacenado | | |
| Precondiciones | Tener un pago, para realizar su respectiva transacción | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Objeto tipo pago, tipo Crédito | | |
| Autor | Juan Posada | | |
| Fecha de creación | 21-11-2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Se genera un pago a realizar | Asigna los atributos del pago |
| 2 | Se realiza el pago por crédito | Verifica la información del usuario para realizar el pago |
| 3 | Finaliza el pago | Pago realizado exitosamente |
| Post condiciones | El sistema registra que el pago fue realizado, con el número de tarjeta correspondiente. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | testTarjetaPago2 | | |
| Identificador caso de prueba | CP0002\_testTarjetaPago2 | | |
| Función probar | Asignación de números de tarjeta de crédito a los pagos | | |
| Objetivo | Verificar que el número de tarjeta de crédito ingresado sea el correcto | | |
| Descripción | Al momento de realizarse un pago con tarjeta de crédito, uno de los atributos más importantes del pago por este método es este. | | |
| Criterios de éxito | El número de tarjeta ingresado coincide con el almacenado | | |
| Criterios de falla | El número de tarjeta difiere con el almacenado | | |
| Precondiciones | Tener un pago, para realizar su respectiva transacción | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Objeto tipo pago, tipo Crédito | | |
| Autor | Juan Posada | | |
| Fecha de creación | 21-11-2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Se genera un pago a realizar | Asigna los atributos del pago |
| 2 | Se realiza el pago por crédito | Verifica la información del usuario para realizar el pago |
| 3 | Finaliza el pago | Pago realizado exitosamente |
| Post condiciones | El sistema registra que el pago fue realizado, con el número de tarjeta correspondiente. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | testTarjetaPago3 | | |
| Identificador caso de prueba | CP0003\_testTarjetaPago3 | | |
| Función probar | Asignación de números de tarjeta de crédito a los pagos | | |
| Objetivo | Verificar que el número de tarjeta de crédito ingresado sea el correcto | | |
| Descripción | Al momento de realizarse un pago con tarjeta de crédito, uno de los atributos más importantes del pago por este método es este. | | |
| Criterios de éxito | El número de tarjeta ingresado coincide con el almacenado | | |
| Criterios de falla | El número de tarjeta difiere con el almacenado | | |
| Precondiciones | Tener un pago, para realizar su respectiva transacción | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Objeto tipo pago, tipo Crédito | | |
| Autor | Juan Posada | | |
| Fecha de creación | 21-11-2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Se genera un pago a realizar | Asigna los atributos del pago |
| 2 | Se realiza el pago por crédito | Verifica la información del usuario para realizar el pago |
| 3 | Finaliza el pago | Pago realizado exitosamente |
| Post condiciones | El sistema registra que el pago fue realizado, con el número de tarjeta correspondiente. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | testSeguCodPago1 | | |
| Identificador caso de prueba | CP0001\_testSeguCodPago1 | | |
| Función probar | Asignación de código de seguridad a un pago por crédito al pago | | |
| Objetivo | Verificar que se asigne el código de seguridad correcto | | |
| Descripción | Verificar que se asigne el código de seguridad correspondiente a la tarjeta de crédito ingresada | | |
| Criterios de éxito | Código de seguridad actual coincide con el código almacenado | | |
| Criterios de falla | Código de seguridad actual difiere del código almacenado | | |
| Precondiciones | Tener un pago, para realizar su respectiva transacción | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Objeto tipo pago, tipo Crédito | | |
| Autor | Juan Posada | | |
| Fecha de creación | 21-11-2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Se genera un pago a realizar | Asigna los atributos del pago |
| 2 | Se realiza el pago por crédito | Verifica la información del usuario para realizar el pago |
| 3 | Finaliza el pago | Pago realizado exitosamente |
| Post condiciones | El sistema registra que el pago fue realizado, con el código de seguridad correspondiente. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | testSeguCodPago2 | | |
| Identificador caso de prueba | CP0002\_testSeguCodPago2 | | |
| Función probar | Asignación de codigo de seguridad a un pago por crédito al pago | | |
| Objetivo | Verificar que se asigne el código de seguridad correcto | | |
| Descripción | Verificar que se asigne el código de seguridad correspondiente a la tarjeta de crédito ingresada | | |
| Criterios de éxito | Código de seguridad actual coincide con el código almacenado | | |
| Criterios de falla | Código de seguridad actual difiere del código almacenado | | |
| Precondiciones | Tener un pago, para realizar su respectiva transacción | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Objeto tipo pago, tipo Crédito | | |
| Autor | Juan Posada | | |
| Fecha de creación | 21-11-2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Se genera un pago a realizar | Asigna los atributos del pago |
| 2 | Se realiza el pago por crédito | Verifica la información del usuario para realizar el pago |
| 3 | Finaliza el pago | Pago realizado exitosamente |
| Post condiciones | El sistema registra que el pago fue realizado, con el código de seguridad correspondiente. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | testSeguCodPago3 | | |
| Identificador caso de prueba | CP0003\_testSeguCodPago3 | | |
| Función probar | Asignación de codigo de seguridad a un pago por crédito al pago | | |
| Objetivo | Verificar que se asigne el código de seguridad correcto | | |
| Descripción | Verificar que se asigne el código de seguridad correspondiente a la tarjeta de crédito ingresada | | |
| Criterios de éxito | Código de seguridad actual coincide con el código almacenado | | |
| Criterios de falla | Código de seguridad actual difiere del código almacenado | | |
| Precondiciones | Tener un pago, para realizar su respectiva transacción | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Objeto tipo pago, tipo Crédito | | |
| Autor | Juan Posada | | |
| Fecha de creación | 21-11-2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Se genera un pago a realizar | Asigna los atributos del pago |
| 2 | Se realiza el pago por crédito | Verifica la información del usuario para realizar el pago |
| 3 | Finaliza el pago | Pago realizado exitosamente |
| Post condiciones | El sistema registra que el pago fue realizado, con el código de seguridad correspondiente. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | testCuotasPago1 | | |
| Identificador caso de prueba | CP0001\_testCuotasPago1 | | |
| Función probar | Asignación del número de cuotas a un pago por crédito al pago | | |
| Objetivo | Verificar que se asigne el número de cuotas correcto | | |
| Descripción | Verificar que se asigne el número de cuotas correspondiente a la tarjeta de crédito ingresada | | |
| Criterios de éxito | número de cuotas actual coincide con el almacenado | | |
| Criterios de falla | número de cuotas actual difiere del almacenado | | |
| Precondiciones | Tener un pago, para realizar su respectiva transacción | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Objeto tipo pago, tipo Crédito | | |
| Autor | Juan Posada | | |
| Fecha de creación | 21-11-2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Se genera un pago a realizar | Asigna los atributos del pago |
| 2 | Se realiza el pago por crédito | Verifica la información del usuario para realizar el pago |
| 3 | Finaliza el pago | Pago realizado exitosamente |
| Post condiciones | El sistema registra que el pago fue realizado, con el número de cuotas correspondiente. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | testCuotasPago2 | | |
| Identificador caso de prueba | CP0002\_testCuotasPago2 | | |
| Función probar | Asignación del número de cuotas a un pago por crédito al pago | | |
| Objetivo | Verificar que se asigne el número de cuotas correcto | | |
| Descripción | Verificar que se asigne el número de cuotas correspondiente a la tarjeta de crédito ingresada | | |
| Criterios de éxito | número de cuotas actual coincide con el almacenado | | |
| Criterios de falla | número de cuotas actual difiere del almacenado | | |
| Precondiciones | Tener un pago, para realizar su respectiva transacción | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Objeto tipo pago, tipo Crédito | | |
| Autor | Juan Posada | | |
| Fecha de creación | 21-11-2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Se genera un pago a realizar | Asigna los atributos del pago |
| 2 | Se realiza el pago por crédito | Verifica la información del usuario para realizar el pago |
| 3 | Finaliza el pago | Pago realizado exitosamente |
| Post condiciones | El sistema registra que el pago fue realizado, con el número de cuotas correspondiente. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | testCuotasPago3 | | |
| Identificador caso de prueba | CP0003\_testCuotasPago3 | | |
| Función probar | Asignación del número de cuotas a un pago por crédito al pago | | |
| Objetivo | Verificar que se asigne el número de cuotas correcto | | |
| Descripción | Verificar que se asigne el número de cuotas correspondiente a la tarjeta de crédito ingresada | | |
| Criterios de éxito | número de cuotas actual coincide con el almacenado | | |
| Criterios de falla | número de cuotas actual difiere del almacenado | | |
| Precondiciones | Tener un pago, para realizar su respectiva transacción | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Objeto tipo pago, tipo Crédito | | |
| Autor | Juan Posada | | |
| Fecha de creación | 21-11-2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Se genera un pago a realizar | Asigna los atributos del pago |
| 2 | Se realiza el pago por crédito | Verifica la información del usuario para realizar el pago |
| 3 | Finaliza el pago | Pago realizado exitosamente |
| Post condiciones | El sistema registra que el pago fue realizado, con el número de cuotas correspondiente. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | testFechaPago7 | | |
| Identificador caso de prueba | CP0007\_testFechaPago7 | | |
| Función probar | Asignación de fecha al momento de realizar un pago por débito | | |
| Objetivo | Asignar la fecha en la que un pago por débito fue realizado | | |
| Descripción | Todo pago al realizarse, genera automáticamente la fecha en la que fue realizado | | |
| Criterios de éxito | La fecha de pago actual coincide con la que se encuentra almacenada | | |
| Criterios de falla | La fecha de pago difiere con la que se encuentra almacenada | | |
| Precondiciones | Tener un pago, para realizar su respectiva transacción | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Objeto tipo pago, tipo Debito | | |
| Autor | Juan Posada | | |
| Fecha de creación | 21-11-2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Se genera un pago a realizar | Asigna los atributos del pago |
| 2 | Se realiza el pago por débito | Transacción realizada |
| 3 | Se le asigna su fecha al pago | Generación automática de la fecha en que se realizó la transacción |
| 4 | Finaliza el pago | Pago realizado exitosamente |
| Post condiciones | El sistema registra que el pago fue realizado, en la fecha correspondiente. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | testFechaPago8 | | |
| Identificador caso de prueba | CP0008\_testFechaPago8 | | |
| Función probar | Asignación de fecha al momento de realizar un pago por débito | | |
| Objetivo | Asignar la fecha en la que un pago por débito fue realizado | | |
| Descripción | Todo pago al realizarse, genera automáticamente la fecha en la que fue realizado | | |
| Criterios de éxito | La fecha de pago actual coincide con la que se encuentra almacenada | | |
| Criterios de falla | La fecha de pago difiere con la que se encuentra almacenada | | |
| Precondiciones | Tener un pago, para realizar su respectiva transacción | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Objeto tipo pago, tipo Efectivo | | |
| Autor | Juan Posada | | |
| Fecha de creación | 21-11-2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Se genera un pago a realizar | Asigna los atributos del pago |
| 2 | Se realiza el pago por débito | Transacción realizada |
| 3 | Se le asigna su fecha al pago | Generación automática de la fecha en que se realizó la transacción |
| 4 | Finaliza el pago | Pago realizado exitosamente |
| Post condiciones | El sistema registra que el pago fue realizado, en la fecha correspondiente. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | testFechaPago9 | | |
| Identificador caso de prueba | CP0009\_testFechaPago9 | | |
| Función probar | Asignación de fecha al momento de realizar un pago por débito | | |
| Objetivo | Asignar la fecha en la que un pago por débito fue realizado | | |
| Descripción | Todo pago al realizarse, genera automáticamente la fecha en la que fue realizado | | |
| Criterios de éxito | La fecha de pago actual coincide con la que se encuentra almacenada | | |
| Criterios de falla | La fecha de pago difiere con la que se encuentra almacenada | | |
| Precondiciones | Tener un pago, para realizar su respectiva transacción | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Objeto tipo pago, tipo Efectivo | | |
| Autor | Juan Posada | | |
| Fecha de creación | 21-11-2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Se genera un pago a realizar | Asigna los atributos del pago |
| 2 | Se realiza el pago por débito | Transacción realizada |
| 3 | Se le asigna su fecha al pago | Generación automática de la fecha en que se realizó la transacción |
| 4 | Finaliza el pago | Pago realizado exitosamente |
| Post condiciones | El sistema registra que el pago fue realizado, en la fecha correspondiente. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | testCuentaPago1 | | |
| Identificador caso de prueba | CP0001\_testCuentaPago1 | | |
| Función probar | Asignación de números de cuenta débito a los pagos | | |
| Objetivo | Verificar que el número de cuenta débito ingresado sea el correcto | | |
| Descripción | Al momento de realizarse un pago con cuenta débito, uno de los atributos más importantes del pago por este método es este. | | |
| Criterios de éxito | El número de cuenta ingresado coincide con el almacenado | | |
| Criterios de falla | El número de cuenta difiere con el almacenado | | |
| Precondiciones | Tener un pago, para realizar su respectiva transacción | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Objeto tipo pago, tipo Débito | | |
| Autor | Juan Posada | | |
| Fecha de creación | 21-11-2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Se genera un pago a realizar | Asigna los atributos del pago |
| 2 | Se realiza el pago por débito | Verifica la información del usuario para realizar el pago |
| 3 | Finaliza el pago | Pago realizado exitosamente |
| Post condiciones | El sistema registra que el pago fue realizado, con el número de cuenta correspondiente. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | testCuentaPago2 | | |
| Identificador caso de prueba | CP0002\_testCuentaPago2 | | |
| Función probar | Asignación de números de cuenta débito a los pagos | | |
| Objetivo | Verificar que el número de cuenta débito ingresado sea el correcto | | |
| Descripción | Al momento de realizarse un pago con cuenta débito, uno de los atributos más importantes del pago por este método es este. | | |
| Criterios de éxito | El número de cuenta ingresado coincide con el almacenado | | |
| Criterios de falla | El número de cuenta difiere con el almacenado | | |
| Precondiciones | Tener un pago, para realizar su respectiva transacción | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Objeto tipo pago, tipo Débito | | |
| Autor | Juan Posada | | |
| Fecha de creación | 21-11-2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Se genera un pago a realizar | Asigna los atributos del pago |
| 2 | Se realiza el pago por débito | Verifica la información del usuario para realizar el pago |
| 3 | Finaliza el pago | Pago realizado exitosamente |
| Post condiciones | El sistema registra que el pago fue realizado, con el número de cuenta correspondiente. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | testCuentaPago3 | | |
| Identificador caso de prueba | CP0003\_testCuentaPago3 | | |
| Función probar | Asignación de números de cuenta débito a los pagos | | |
| Objetivo | Verificar que el número de cuenta débito ingresado sea el correcto | | |
| Descripción | Al momento de realizarse un pago con cuenta débito, uno de los atributos más importantes del pago por este método es este. | | |
| Criterios de éxito | El número de cuenta ingresado coincide con el almacenado | | |
| Criterios de falla | El número de cuenta difiere con el almacenado | | |
| Precondiciones | Tener un pago, para realizar su respectiva transacción | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Objeto tipo pago, tipo Débito | | |
| Autor | Juan Posada | | |
| Fecha de creación | 21-11-2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Se genera un pago a realizar | Asigna los atributos del pago |
| 2 | Se realiza el pago por débito | Verifica la información del usuario para realizar el pago |
| 3 | Finaliza el pago | Pago realizado exitosamente |
| Post condiciones | El sistema registra que el pago fue realizado, con el número de cuenta correspondiente. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | pagoEfectivoTest | | |
| Identificador caso de prueba | CP001\_PagoEfectivoTest | | |
| Función probar | El usuario tiene una sesión activa y se registra el pago en el arreglo de pagos | | |
| Objetivo | Verificar que la sesión dada por el usuario está registrada en el arreglo y el pago queda registrado en el arreglo. | | |
| Descripción | Probar que el pago se realiza con éxito. | | |
| Criterios de éxito | El pago queda registrado en el arreglo y el número de referencia guardado es el mismo que el ingresado. | | |
| Criterios de falla | El pago no queda registrado, el número de referencia no coincide con el ingresado. | | |
| Precondiciones | Sesión guardada en el arreglo de sesiones | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Datos requeridos para realizar el pago en efectivo (Valor, Referencia, ID usuario que paga, ID usuario a pagar, concepto del pago, fecha) | | |
| Autor | Santiago Cepeda | | |
| Fecha de creación | 21 de noviembre 2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Se guarda la sesión en el arreglo | Retorna la sesión guardada |
| 2 | Se ejecuta el método de registrar el pago en efectivo con sus datos | Retorna el pago con todos sus atributos y se guarda en el sistema |
|  |  |  |
| Post condiciones | El pago queda registrado para ser consultado posteriormente | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | pagoEfectivoTest2 | | |
| Identificador caso de prueba | CP001\_PagoEfectivoTest2 | | |
| Función probar | El usuario tiene una sesión activa y se registra el pago en el arreglo de pagos | | |
| Objetivo | Verificar que la sesión dada por el usuario está registrada en el arreglo y el pago queda registrado en el arreglo. | | |
| Descripción | Probar que el pago se realiza con éxito. | | |
| Criterios de éxito | El pago queda registrado en el arreglo y el número de ID de el usuario pagado guardado es el mismo que el ingresado. | | |
| Criterios de falla | El pago no queda registrado, el número de ID del usuario pagado no coincide con el ingresado. | | |
| Precondiciones | Sesión guardada en el arreglo de sesiones | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Datos requeridos para realizar el pago en efectivo (Valor, Referencia, ID usuario que paga, ID usuario a pagar, concepto del pago, fecha) | | |
| Autor | Santiago Cepeda | | |
| Fecha de creación | 21 de noviembre 2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Se guarda la sesión en el arreglo | Retorna la sesión guardada |
| 2 | Se ejecuta el método de registrar el pago en efectivo con sus datos | Retorna el pago con todos sus atributos y se guarda en el sistema |
|  |  |  |
| Post condiciones | El pago queda registrado para ser consultado posteriormente | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | pagoEfectivoTest3 | | |
| Identificador caso de prueba | CP001\_PagoEfectivoTest | | |
| Función probar | El usuario que no tiene sesión activa no puede registrar pagos en el sistema | | |
| Objetivo | Verificar que la sesión dada por el usuario no está registrada en el arreglo y el pago no queda registrado en el arreglo. | | |
| Descripción | Probar que el pago no se realiza con éxito. | | |
| Criterios de éxito | El pago no queda registrado en el arreglo. | | |
| Criterios de falla | El pago queda registrado. | | |
| Precondiciones | Sesión que no esté guardada en el arreglo de sesiones. | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Datos requeridos para realizar el pago en efectivo (Valor, Referencia, ID usuario que paga, ID usuario a pagar, concepto del pago, fecha) | | |
| Autor | Santiago Cepeda | | |
| Fecha de creación | 21 de noviembre 2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Se ejecuta el método de registrar el pago en efectivo con sus datos | Retorna un mensaje de error diciendo que no se pudo realizar el pago |
|  |  |  |
|  |  |  |
| Post condiciones | El pago no queda registrado en el sistema | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | pagoDebitoTest | | |
| Identificador caso de prueba | CP001\_PagoDebitoTest | | |
| Función probar | El usuario tiene una sesión activa y se registra el pago en el arreglo de pagos | | |
| Objetivo | Verificar que la sesión dada por el usuario está registrada en el arreglo y el pago queda registrado en el arreglo. | | |
| Descripción | Probar que el pago se realiza con éxito. | | |
| Criterios de éxito | El pago queda registrado en el arreglo y el número de referencia guardado es el mismo que el ingresado. | | |
| Criterios de falla | El pago no queda registrado, el número de referencia no coincide con el ingresado. | | |
| Precondiciones | Sesión guardada en el arreglo de sesiones | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Datos requeridos para realizar el pago por cuenta debito (Valor, Referencia, ID usuario que paga, ID usuario a pagar, concepto del pago,fecha, número de cuenta) | | |
| Autor | Santiago Cepeda | | |
| Fecha de creación | 21 de noviembre 2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Se guarda la sesión en el arreglo | Retorna la sesión guardada |
| 2 | Se ejecuta el método de registrar el pago por débito con sus datos | Retorna el pago con todos sus atributos y se guarda en el sistema |
|  |  |  |
| Post condiciones | El pago queda registrado para ser consultado posteriormente | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | pagoDebitoTest2 | | |
| Identificador caso de prueba | CP001\_PagoDebitoTest2 | | |
| Función probar | El usuario tiene una sesión activa y se registra el pago en el arreglo de pagos | | |
| Objetivo | Verificar que la sesión dada por el usuario está registrada en el arreglo y el pago queda registrado en el arreglo. | | |
| Descripción | Probar que el pago se realiza con éxito. | | |
| Criterios de éxito | El pago queda registrado en el arreglo y el número de ID de el usuario que paga guardado es el mismo que el ingresado. | | |
| Criterios de falla | El pago no queda registrado, el número de ID del usuario que paga no coincide con el ingresado. | | |
| Precondiciones | Sesión guardada en el arreglo de sesiones | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Datos requeridos para realizar el pago por cuenta debito (Valor, Referencia, ID usuario que paga, ID usuario a pagar, concepto del pago,fecha, número de cuenta) | | |
| Autor | Santiago Cepeda | | |
| Fecha de creación | 21 de noviembre 2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Se guarda la sesión en el arreglo | Retorna la sesión guardada |
| 2 | Se ejecuta el método de registrar el pago por débito con sus datos | Retorna el pago con todos sus atributos y se guarda en el sistema |
|  |  |  |
| Post condiciones | El pago queda registrado para ser consultado posteriormente | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | pagoDebitoTest3 | | |
| Identificador caso de prueba | CP001\_PagoDebitoTest3 | | |
| Función probar | El usuario que no tiene sesión activa no puede registrar pagos en el sistema | | |
| Objetivo | Verificar que la sesión dada por el usuario no está registrada en el arreglo y el pago no queda registrado en el arreglo. | | |
| Descripción | Probar que el pago no se realiza con éxito. | | |
| Criterios de éxito | El pago no queda registrado en el arreglo. | | |
| Criterios de falla | El pago queda registrado. | | |
| Precondiciones | Sesión que no esté guardada en el arreglo de sesiones. | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Datos requeridos para realizar el pago por cuenta debito (Valor, Referencia, ID usuario que paga, ID usuario a pagar, concepto del pago, fecha, número de cuenta) | | |
| Autor | Santiago Cepeda | | |
| Fecha de creación | 21 de noviembre 2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Se ejecuta el método de registrar el pago por débito con sus datos | Retorna un mensaje de error diciendo que no se pudo realizar el pago |
|  |  |  |
|  |  |  |
| Post condiciones | El pago no queda almacenado en el sistema. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | pagoCreditoTest | | |
| Identificador caso de prueba | CP001\_PagoCreditoTest | | |
| Función probar | El usuario tiene una sesión activa y se registra el pago en el arreglo de pagos | | |
| Objetivo | Verificar que la sesión dada por el usuario está registrada en el arreglo y el pago queda registrado en el arreglo. | | |
| Descripción | Probar que el pago se realiza con éxito. | | |
| Criterios de éxito | El pago queda registrado en el arreglo y el número de referencia guardado es el mismo que el ingresado. | | |
| Criterios de falla | El pago no queda registrado, el número de referencia no coincide con el ingresado. | | |
| Precondiciones | Sesión guardada en el arreglo de sesiones | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Datos requeridos para realizar el pago por tarjeta crédito (Valor, Referencia, ID usuario que paga, ID usuario a pagar, concepto del pago,fecha, número de tarjeta, código de seguridad, cuotas) | | |
| Autor | Santiago Cepeda | | |
| Fecha de creación | 21 de noviembre 2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Se guarda la sesión en el arreglo | Retorna la sesión guardada |
| 2 | Se ejecuta el método de registrar el pago por Crédito con sus datos | Retorna el pago con todos sus atributos y se guarda en el sistema |
|  |  |  |
| Post condiciones | El pago queda registrado para ser consultado posteriormente | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | PagoCreditoTest2 | | |
| Identificador caso de prueba | CP001\_PagoCreditoTest2 | | |
| Función probar | El usuario tiene una sesión activa y se registra el pago en el arreglo de pagos | | |
| Objetivo | Verificar que la sesión dada por el usuario está registrada en el arreglo y el pago queda registrado en el arreglo. | | |
| Descripción | Probar que el pago se realiza con éxito. | | |
| Criterios de éxito | El pago queda registrado en el arreglo y el número de ID de el usuario pagado guardado es el mismo que el ingresado. | | |
| Criterios de falla | El pago no queda registrado, el número de ID del usuario pagado no coincide con el ingresado. | | |
| Precondiciones | Sesión guardada en el arreglo de sesiones | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Datos requeridos para realizar el pago por tarjeta crédito (Valor, Referencia, ID usuario que paga, ID usuario a pagar, concepto del pago,fecha, número de tarjeta, código de seguridad, cuotas) | | |
| Autor | Santiago Cepeda | | |
| Fecha de creación | 21 de noviembre 2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Se guarda la sesión en el arreglo | Retorna la sesión guardada |
| 2 | Se ejecuta el método de registrar el pago por Crédito con sus datos | Retorna el pago con todos sus atributos y se guarda en el sistema |
|  |  |  |
| Post condiciones | El pago queda registrado para ser consultado posteriormente | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | pagoCreditoTest3 | | |
| Identificador caso de prueba | CP001\_PagoCreditoTest3 | | |
| Función probar | El usuario que no tiene sesión activa no puede registrar pagos en el sistema | | |
| Objetivo | Verificar que la sesión dada por el usuario no está registrada en el arreglo y el pago no queda registrado en el arreglo. | | |
| Descripción | Probar que el pago no se realiza con éxito. | | |
| Criterios de éxito | El pago no queda registrado en el arreglo. | | |
| Criterios de falla | El pago queda registrado. | | |
| Precondiciones | Sesión que no esté guardada en el arreglo de sesiones. | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Datos requeridos para realizar el pago por tarjeta de crédito (Valor, Referencia, ID usuario que paga, ID usuario a pagar, concepto del pago, fecha, número de tarjeta, código de seguridad, cuotas) | | |
| Autor | Santiago Cepeda | | |
| Fecha de creación | 21 de noviembre 2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Se ejecuta el método de registrar el pago por Crédito con sus datos | Retorna un mensaje de error diciendo que no se pudo realizar el pago |
|  |  |  |
|  |  |  |
| Post condiciones | El pago no queda almacenado en el sistema. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | pagoSEfectivo | | |
| Identificador caso de prueba | CP001\_PagoSEfectivo | | |
| Función probar | Inicio de sesion, verificación de sesion, realizar pago en efectivo | | |
| Objetivo | Verificar el correcto funcionamiento de la funcionalidad de realizar pago en efectivo desde el inicio de sesión | | |
| Descripción | Prueba de sistema para el pago en efectivo desde el inicio de sesión | | |
| Criterios de éxito | El usuario puede registrar el pago en efectivo correctamente y el número de referencia ingresado y el guardado es el mismo. | | |
| Criterios de falla | No se realiza el pago | | |
| Precondiciones | El usuario existe en el sistema | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Usuario creado, credenciales del usuario correctos, datos del pago en efectivo a registrar (Valor, Referencia, ID usuario que paga, ID usuario a pagar, concepto del pago, fecha) | | |
| Autor | Julian Ballesteros-Santiago Cepeda-Juan Posada | | |
| Fecha de creación | 21 noviembre 2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | El usuario inicia sesión | numero de sesion , sesion guardada en el arreglo |
| 2 | el usuario ejecuta el método de pagar en efectivo | información del pago y se guarda el pago |
|  |  |  |
| Post condiciones | La sesión queda registrada en el sistema, el pago queda registrado en el sistema | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | PagoSEfectivo2 | | |
| Identificador caso de prueba | CP001\_PagoSEfectivo2 | | |
| Función probar | Usuario con credenciales incorrectos no puede iniciar sesión por lo tanto no puede registrar un pago | | |
| Objetivo | Verificar el correcto funcionamiento de la funcionalidad verificación de usuarios para evitar que usuarios sin sesión activa registren pagos. | | |
| Descripción | Prueba de sistema para no permitir pago en efectivo sin sesión activa. | | |
| Criterios de éxito | El usuario puede no registrar el pago en efectivo correctamente. | | |
| Criterios de falla | El usuario puede registrar el pago en efectivo correctamente | | |
| Precondiciones | El usuario no existe en el sistema o no ingreso credenciales válidos | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Usuario no creado, credenciales del usuario incorrectos, datos del pago en efectivo a registrar (Valor, Referencia, ID usuario que paga, ID usuario a pagar, concepto del pago, fecha) | | |
| Autor | Julian Ballesteros-Santiago Cepeda-Juan Posada | | |
| Fecha de creación | 21 noviembre 2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | El usuario inicia sesión | Error usuario no encontrado |
| 2 | el usuario ejecuta el método de pagar en efectivo | No tiene sesión activa error al registrar el pago |
|  |  |  |
| Post condiciones | El pago no queda registrado en el sistema. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | pagoSDebito | | |
| Identificador caso de prueba | CP001\_PagoSDebito | | |
| Función probar | Inicio de sesion, verificación de sesion, realizar pago por cuenta débito | | |
| Objetivo | Verificar el correcto funcionamiento de la funcionalidad de realizar pago por cuenta debito desde el inicio de sesión | | |
| Descripción | Prueba de sistema para el pago por cuenta debito desde el inicio de sesión | | |
| Criterios de éxito | El usuario puede registrar el pago por cuenta débito correctamente y el número de ID de el usuario que paga ingresado y el guardado es el mismo. | | |
| Criterios de falla | No se realiza el pago | | |
| Precondiciones | El usuario existe en el sistema | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Usuario creado, credenciales del usuario correctos, datos del pago por cuenta debito a registrar (Valor, Referencia, ID usuario que paga, ID usuario a pagar, concepto del pago, fecha, número de cuenta) | | |
| Autor | Julian Ballesteros-Santiago Cepeda-Juan Posada | | |
| Fecha de creación | 21 noviembre 2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | El usuario inicia sesión | numero de sesion , sesion guardada en el arreglo |
| 2 | el usuario ejecuta el método de pagar por cuenta débito | información del pago y se guarda el pago |
|  |  |  |
| Post condiciones | La sesión queda registrada en el sistema, el pago queda registrado en el sistema | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | pagoSDebito2 | | |
| Identificador caso de prueba | CP001\_PagoSDebito2 | | |
| Función probar | Usuario con credenciales incorrectos no puede iniciar sesión por lo tanto no puede registrar un pago | | |
| Objetivo | Verificar el correcto funcionamiento de la funcionalidad verificación de usuarios para evitar que usuarios sin sesión activa registren pagos. | | |
| Descripción | Prueba de sistema para no permitir pago por cuenta débito sin sesión activa. | | |
| Criterios de éxito | El usuario puede no registrar el pago correctamente. | | |
| Criterios de falla | El usuario puede registrar el pago por cuenta débito correctamente | | |
| Precondiciones | El usuario no existe en el sistema o no ingreso credenciales válidos | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Usuario no creado, credenciales del usuario incorrectos, datos del pago por cuenta debito a registrar (Valor, Referencia, ID usuario que paga, ID usuario a pagar, concepto del pago, fecha, número de cuenta) | | |
| Autor | Julian Ballesteros-Santiago Cepeda-Juan Posada | | |
| Fecha de creación | 21 noviembre 2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | El usuario inicia sesión | Error usuario no encontrado |
| 2 | el usuario ejecuta el método de pagar por cuenta débito | No tiene sesión activa error al registrar el pago |
|  |  |  |
| Post condiciones | El pago no queda registrado en el sistema. | | |

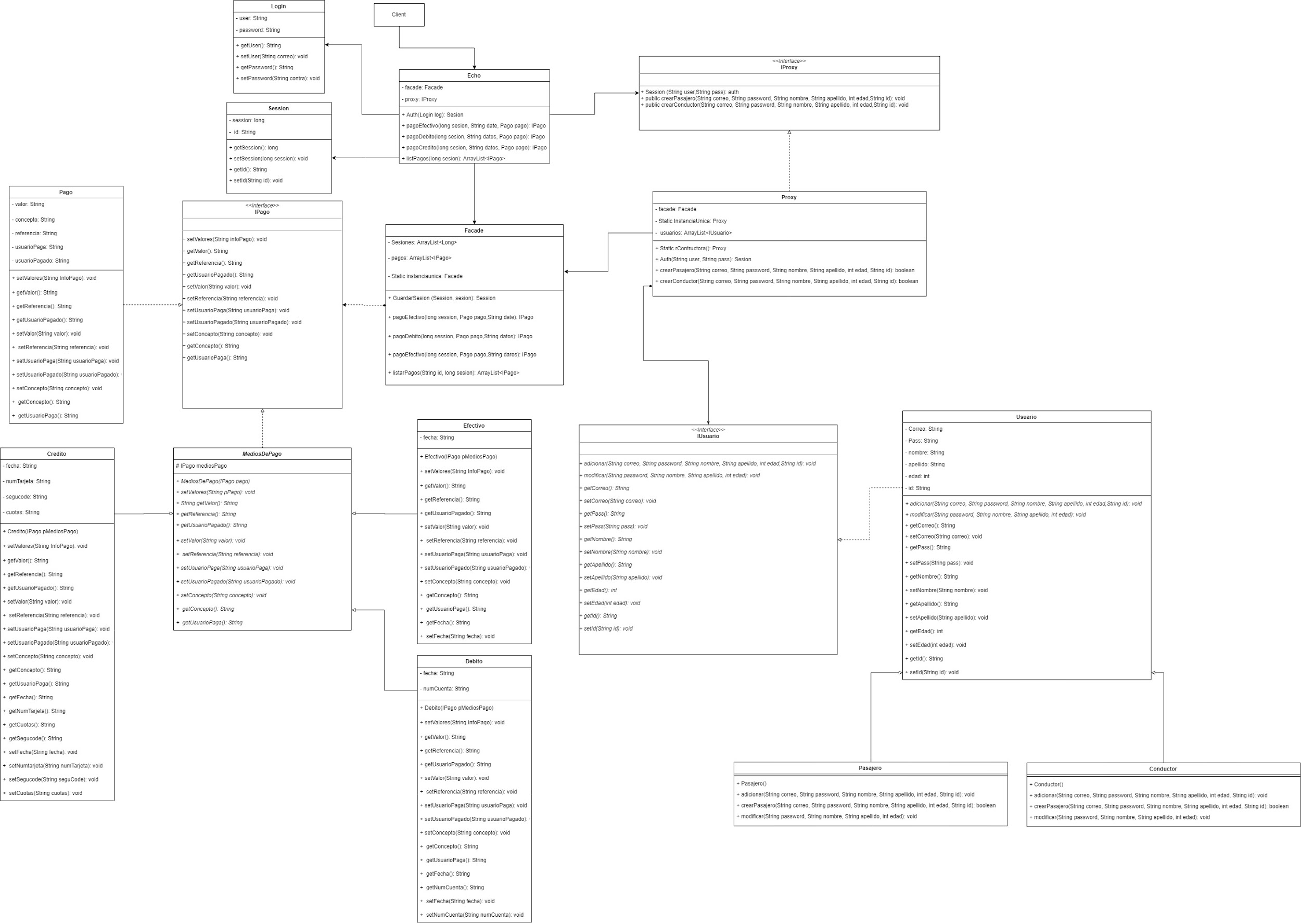
|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | pagoSCredito | | |
| Identificador caso de prueba | CP001\_PagoSCredito | | |
| Función probar | Inicio de sesion, verificación de sesion, realizar pago por cuenta débito | | |
| Objetivo | Verificar el correcto funcionamiento de la funcionalidad de realizar pago por Tarjeta crédito desde el inicio de sesión | | |
| Descripción | Prueba de sistema para el pago por tarjeta crédito desde el inicio de sesión | | |
| Criterios de éxito | El usuario puede registrar el pago por tarjeta crédito correctamente y el número de ID de el usuario pagado ingresado y el guardado es el mismo. | | |
| Criterios de falla | No se realiza el pago | | |
| Caso de prueba | pagoSCredito | | |
| Precondiciones | El usuario existe en el sistema | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Usuario creado, credenciales del usuario correctos, datos del pago por tarjeta de crédito a registrar (Valor, Referencia, ID usuario que paga, ID usuario a pagar, concepto del pago, fecha, número de tarjeta, código de seguridad, cuotas) | | |
| Autor | Julian Ballesteros-Santiago Cepeda-Juan Posada | | |
| Fecha de creación | 21 noviembre 2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | El usuario inicia sesión | numero de sesion , sesion guardada en el arreglo |
| 2 | el usuario ejecuta el método de pagar por tarjeta crédito | información del pago y se guarda el pago |
|  |  |  |
| Post condiciones | La sesión queda registrada en el sistema, el pago queda registrado en el sistema | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Identificador caso de prueba | CP001\_PagoSCredito2 | | |
| Función probar | Usuario con credenciales incorrectos no puede iniciar sesión por lo tanto no puede registrar un pago | | |
| Objetivo | Verificar el correcto funcionamiento de la funcionalidad verificación de usuarios para evitar que usuarios sin sesión activa registren pagos. | | |
| Descripción | Prueba de sistema para no permitir pago por tarjeta crédito sin sesión activa. | | |
| Criterios de éxito | El usuario puede no registrar el pago correctamente. | | |
| Criterios de falla | El usuario puede registrar el pago por tarjeta crédito correctamente | | |
| Precondiciones | El usuario no existe en el sistema o no ingreso credenciales válidos | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Usuario no creado, credenciales del usuario incorrectos, datos del pago por tarjeta crédito a registrar (Valor, Referencia, ID usuario que paga, ID usuario a pagar, concepto del pago, fecha, número de tarjeta, código de seguridad, cuotas) | | |
| Autor | Julian Ballesteros-Santiago Cepeda-Juan Posada | | |
| Fecha de creación | 21 noviembre 2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | El usuario inicia sesión | Error usuario no encontrado |
| 2 | el usuario ejecuta el método de pagar por tarjeta crédito | No tiene sesión activa error al registrar el pago |
|  |  |  |
| Post condiciones | El pago no queda registrado en el sistema. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | pagoSCredito | | |
| Identificador caso de prueba | CP001\_PagoSCredito | | |
| Función probar | Inicio de sesion, verificación de sesion, realizar pago por cuenta débito | | |
| Objetivo | Verificar el correcto funcionamiento de la funcionalidad de realizar pago por Tarjeta crédito desde el inicio de sesión | | |
| Descripción | Prueba de sistema para el pago por tarjeta crédito desde el inicio de sesión | | |
| Criterios de éxito | El usuario puede registrar el pago por tarjeta crédito correctamente y el número de ID de el usuario pagado ingresado y el guardado es el mismo. | | |
| Criterios de falla | No se realiza el pago | | |
| Caso de prueba | pagoSCredito | | |
| Precondiciones | El usuario existe en el sistema | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Usuario creado, credenciales del usuario correctos, datos del pago por tarjeta de crédito a registrar (Valor, Referencia, ID usuario que paga, ID usuario a pagar, concepto del pago, fecha, número de tarjeta, código de seguridad, cuotas) | | |
| Autor | Julian Ballesteros-Santiago Cepeda-Juan Posada | | |
| Fecha de creación | 21 noviembre 2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | El usuario inicia sesión | numero de sesion , sesion guardada en el arreglo |
| 2 | el usuario ejecuta el método de pagar por tarjeta crédito | información del pago y se guarda el pago |
|  |  |  |
| Post condiciones | La sesión queda registrada en el sistema, el pago queda registrado en el sistema | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Caso de prueba | listarPagosTest | | |
| Identificador caso de prueba | CP001\_listarPagosTest | | |
| Función probar | Se guarda un pago correctamente en el arreglo y se retorna todo el arreglo de pagos si la sesión es válida. | | |
| Objetivo | Verificar el correcto funcionamiento de la funcionalidad verificación de sesión para evitar que usuarios sin sesión activa registren y recuperen la lista de pagos. | | |
| Descripción | Prueba de sistema para permitir recuperar lista de pagos si tiene sesión activa. | | |
| Criterios de éxito | Se recupera la lista de pagos | | |
| Criterios de falla | No se obtiene la lista de pagos | | |
| Precondiciones | Sesión guardada en el sistema | | |
| Perfil del usuario |  | | |
| Necesidades para el caso de prueba | Sesion activa | | |
| Autor | Julian Ballesteros-Santiago Cepeda-Juan Posada | | |
| Fecha de creación | 21 noviembre 2018 | | |
| Flujo del caso de prueba | **No paso** | **Usuario del sistema** | **Sistema** |
| 1 | Se ejecuta el método de listar pagos y se envía la sesión | Devuelve la lista de pagos |
| 2 |  |  |
|  |  |  |
| Post condiciones | Se recupera la lista de pagos. | | |

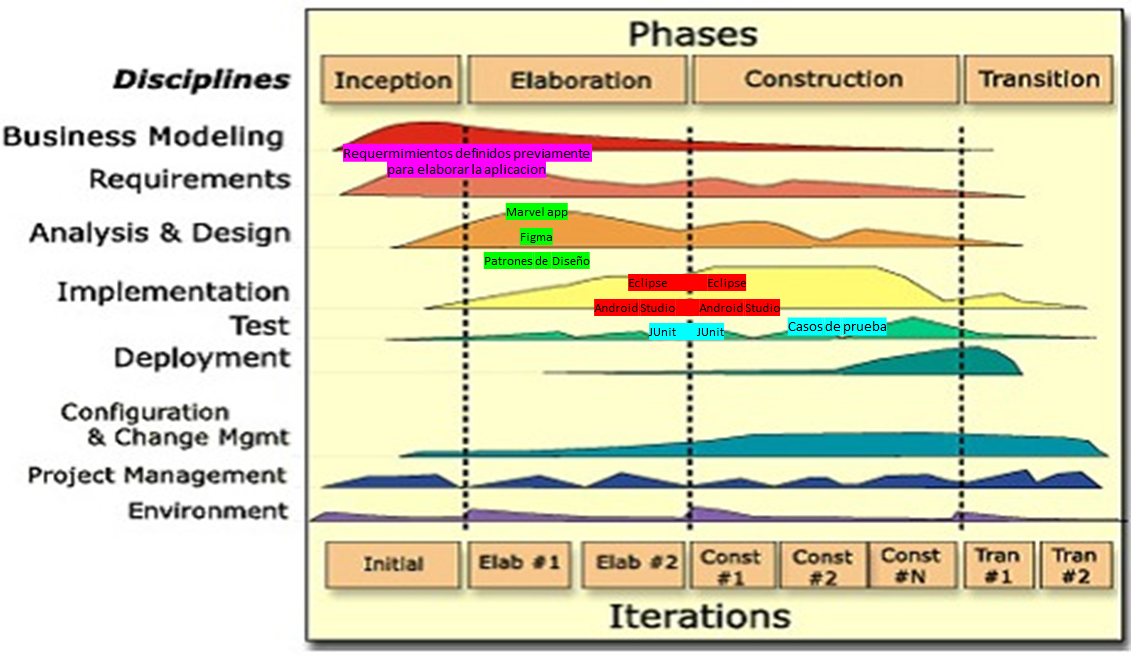
1. ***Diagrama lógico de la arquitectura (basado en el que se dio de base).***

******

1. ***Explicación sobre todos los patrones de diseño de software dados en la vista lógica.***

* Facade, para tener una interfaz funcional y simplificada de todos los distintos procesos que se llevan a cabo en la aplicación de pagos, este maneja todo el sistema de realización de pagos en cualquiera de las modalidades, y el cliente simplemente se tiene que preocupar por enviar los parámetros solicitados. También controla el acceso a dichos métodos por una verificación de número de sesión que el proxy guarda en el facade a el momento de realizar el inicio de sesión.
* Proxy, se utilizó este patrón de diseño para controlar el acceso a las demás funcionalidades del sistema, este tiene un arreglo de usuarios y en el momento de realizar una petición de inicio de sesión si los credenciales son correctos retorna el numero de sesion y guarda este numero en el facade, para poder acceder a las demás funcionalidades
* Decorator, este utilizó este patrón de diseño ya que facilita añadir funcionalidades de manera dinámica lo cual se ajustaba perfecto a lo que se necesita a la hora de los distintos medios de pago, ya que para cada uno de estos se agregan distintos atributos que los diferencia de los demás medios de pago.
* Singleton, lo utilizamos para que el proxy y el facade solo tengan una instancia lo cual nos permite acceder desde distintos dispositivos y clases dentro del sistema y no se creará una nueva instancia por cada uno de estos.

1. ***Gráfica de la metodología unificada de desarrollo y sobre esta nombrar los: estándares, herramientas y metodologías usadas en el flujo de trabajo y en cada fase.***

******

***Los estándares utilizados en la aplicación fueron Rest endpoint***

En la fase de inicio:

Se estudió el proyecto y que posibilidades tenemos de realizarlo en distintas plataformas

Se definieron todos los requisitos funcionales y no funcionales del proyecto, los casos de uso y por último se definieron los software que usamos para llevar a cabo el back end y front end.

Las herramientas usadas para llevar a cabo esta fase fue:

-Excel

-Visual Paradigm.

En la fase de elaboración:

Teniendo todo definido para el proyecto empezamos a desarrollar el backend en eclipse por medio de el google app engine e implementando los patrones de diseño siendo estos Facade,Proxy, Singleton y Decorator, además de Github para tener un control de las versiones. Teniendo en cuenta que antes se había planeado el uso de Appi´s para los pagos pero debido a que no se pudo implementar, se usó el patrón de diseño Decorator para añadir pagos.

Por otro lado se uso la herramienta MarvelApp para tomar como referencia las pantallas de la aplicación, luego estas mismas se implementaron en la aplicación Figma para poder medio de esta usar su código e integrarlo con Android Studio para hacer las pestañas funcionales.

Las herramientas usadas para esta fase fueron:

-Github

-Eclipse

-Google App Engine

-Figma

-Marvel App

-Android Studio

Fase de construcción:

En esta fase, luego de tener todas las herramientas y empezar a elaborar la aplicación en backend y frontend, se empiezan a programar todas las funcionalidades de la aplicación y se empieza a desarrollar toda la conexión entre el backend y frontend, también para asegurar el funcionamiento de la aplicación se empiezan a desarrollar pruebas para verificar su buen funcionalidad.

Las herramientas usadas para llevar a cabo esta fase fueron:

-Android Studio

-Eclipse

-GoogleApp Engine

-Github

-JUnit

En la fase de transición:

Teniendo ya la conexión entre backend y frontend, además de todas las funcionalidades requeridas terminadas,por último realizamos las pruebas del sistema y realizamos pruebas completas del funcionamiento de la aplicación, para en caso de, realizar los ajustes necesarios para la presentación.

Las herramientas usadas para llevar a cabo esta fase fueron:

-Android Studio

-Eclipse

-JUnit

-Github

1. ***Código de JUNIT para las pruebas de 1 y 2***

***FACADE INTEGRATION TEST***

|  |  |
| --- | --- |
| ***Pacge test;*** |  |
|  |  |
|  | ***import static org.junit.Assert.\*;*** |
|  |  |
|  | ***import org.junit.Test;*** |
|  |  |
|  | ***import com.example.echo.Facade;*** |
|  | ***import com.example.echo.IPago;*** |
|  | ***import com.example.echo.Pago;*** |
|  | ***import com.example.echo.Proxy;*** |
|  | ***import com.example.echo.Session;*** |
|  |  |
|  | ***public class FacadeITest {*** |
|  | ***Facade facade;*** |
|  | ***Session ses;*** |
|  | ***Session prueba;*** |
|  | ***IPago pago;*** |
|  |  |
|  | ***private void setupEscenario1() {*** |
|  | ***facade = Facade.rConstructora();*** |
|  | ***ses = new Session();*** |
|  | ***ses.setSession(38712555);*** |
|  | ***ses.setId("1015678");*** |
|  | ***prueba = facade.guardarSesion(ses);*** |
|  | ***pago = new Pago();*** |
|  | ***pago.setValores("3000,4567655,"+prueba.getId()+",123456,Multa");*** |
|  |  |
|  | ***}*** |
|  | ***private void setupEscenario2() {*** |
|  | ***facade = Facade.rConstructora();*** |
|  | ***ses = new Session();*** |
|  | ***ses.setSession(38712566);*** |
|  | ***ses.setId("1015678");*** |
|  | ***prueba = facade.guardarSesion(ses);*** |
|  | ***pago = new Pago();*** |
|  | ***pago.setValores("3000,4567655,"+prueba.getId()+",123456,Viaje");*** |
|  | ***}*** |
|  |  |
|  |  |
|  | ***@Test*** |
|  | ***public void pagoEfectivoTest() {*** |
|  | ***setupEscenario1();*** |
|  | ***IPago respuesta;*** |
|  | ***try {*** |
|  | ***respuesta = facade.pagoEfectivo(prueba.getSession(),(Pago) pago, "12-12-12");*** |
|  | ***assertEquals("El numero de referencia deberia ser el mismo",pago.getReferencia(),respuesta.getReferencia());*** |
|  | ***} catch (Exception e) {*** |
|  | ***fail("Se debio realizar el pago en Efectivo");*** |
|  | ***}*** |
|  |  |
|  |  |
|  | ***}*** |
|  | ***@Test*** |
|  | ***public void pagoEfectivoTest2() {*** |
|  | ***setupEscenario2();*** |
|  | ***IPago respuesta;*** |
|  | ***try {*** |
|  | ***respuesta = facade.pagoEfectivo(prueba.getSession(),(Pago) pago, "12-12-12");*** |
|  | ***assertEquals("El numero de ID del usuario pagado deberia ser el mismo",pago.getUsuarioPagado(),respuesta.getUsuarioPagado());*** |
|  | ***} catch (Exception e) {*** |
|  | ***fail("Se debio realizar el pago en Efectivo");*** |
|  | ***}*** |
|  | ***}*** |
|  |  |
|  | ***@Test*** |
|  | ***public void pagoEfectivoTest3() {*** |
|  | ***setupEscenario1();*** |
|  | ***IPago respuesta;*** |
|  | ***try {*** |
|  | ***respuesta = facade.pagoEfectivo(38712111,(Pago) pago, "12-12-12");*** |
|  | ***} catch (Exception e) {*** |
|  | ***assertEquals("Error no se pudo realizar el pago en efectivo",e.getMessage());*** |
|  | ***}*** |
|  | ***}*** |
|  | ***@Test*** |
|  | ***public void pagoDebitoTest() {*** |
|  | ***setupEscenario1();*** |
|  | ***IPago respuesta;*** |
|  | ***try {*** |
|  | ***respuesta = facade.pagoDebito(prueba.getSession(),(Pago) pago, "12-12-12,455556543");*** |
|  | ***assertEquals("El numero de referencia deberia ser el mismo",pago.getReferencia(),respuesta.getReferencia());*** |
|  |  |
|  | ***} catch (Exception e) {*** |
|  | ***fail("Se debio realizar el pago por Debito");*** |
|  | ***}*** |
|  |  |
|  | ***}*** |
|  | ***@Test*** |
|  | ***public void pagoDebitoTest2() {*** |
|  | ***setupEscenario2();*** |
|  | ***IPago respuesta;*** |
|  | ***try {*** |
|  | ***respuesta = facade.pagoDebito(prueba.getSession(),(Pago) pago, "12-12-12,455556543");*** |
|  | ***assertEquals("El numero de ID del usuario que paga deberia ser el mismo",pago.getUsuarioPaga(),respuesta.getUsuarioPaga());*** |
|  |  |
|  | ***} catch (Exception e) {*** |
|  | ***fail("Se debio realizar el pago por Debito");*** |
|  | ***}*** |
|  |  |
|  | ***}*** |
|  | ***@Test*** |
|  | ***public void pagoDebitoTest3() {*** |
|  | ***setupEscenario1();*** |
|  | ***IPago respuesta;*** |
|  | ***try {*** |
|  | ***respuesta = facade.pagoDebito(38712111,(Pago) pago, "12-12-12,455556543");*** |
|  | ***} catch (Exception e) {*** |
|  | ***assertEquals("Error no se pudo realizar el pago por debito",e.getMessage());*** |
|  | ***}*** |
|  | ***}*** |
|  | ***@Test*** |
|  | ***public void pagoCreditoTest() {*** |
|  | ***setupEscenario1();*** |
|  | ***IPago respuesta;*** |
|  | ***try {*** |
|  | ***respuesta = facade.pagoCredito(prueba.getSession(),(Pago) pago, "12-12-12,455556543123,433,2");*** |
|  | ***assertEquals("El numero de referencia deberia ser el mismo",pago.getReferencia(),respuesta.getReferencia());*** |
|  | ***} catch (Exception e) {*** |
|  | ***fail("Se debio realizar el pago por Credito");*** |
|  | ***}*** |
|  | ***}*** |
|  | ***@Test*** |
|  | ***public void pagoCreditoTest2() {*** |
|  | ***setupEscenario2();*** |
|  | ***IPago respuesta;*** |
|  | ***try {*** |
|  | ***respuesta = facade.pagoCredito(prueba.getSession(),(Pago) pago, "12-12-12,455556543123,433,2");*** |
|  | ***assertEquals("El numero de ID del usuario pagado deberia ser el mismo",pago.getUsuarioPaga(),respuesta.getUsuarioPaga());*** |
|  | ***} catch (Exception e) {*** |
|  | ***fail("Se debio realizar el pago por Credito");*** |
|  | ***}*** |
|  | ***}*** |
|  | ***@Test*** |
|  | ***public void pagoCreditoTest3() {*** |
|  | ***setupEscenario1();*** |
|  | ***IPago respuesta;*** |
|  | ***try {*** |
|  | ***respuesta = facade.pagoCredito(38712111,(Pago) pago, "12-12-12,455556543123,433,2");*** |
|  | ***} catch (Exception e) {*** |
|  | ***assertEquals("Error no se pudo realizar el pago por credito",e.getMessage());*** |
|  | ***}*** |
|  | ***}*** |
|  |  |
|  | ***}*** |

***Facade test unitario***

|  |  |
| --- | --- |
| ***package test;*** |  |
|  |  |
|  | ***import static org.junit.Assert.\*;*** |
|  |  |
|  | ***import org.junit.Test;*** |
|  |  |
|  | ***import com.example.echo.Debito;*** |
|  | ***import com.example.echo.Efectivo;*** |
|  | ***import com.example.echo.Facade;*** |
|  | ***import com.example.echo.IPago;*** |
|  | ***import com.example.echo.Pago;*** |
|  | ***import com.example.echo.Session;*** |
|  |  |
|  | ***public class FacadeUTest {*** |
|  |  |
|  | ***@Test*** |
|  | ***public void guardarSesionesTest() {*** |
|  | ***Facade facade = Facade.rConstructora();*** |
|  | ***Session ses = new Session();*** |
|  | ***ses.setSession(38712555);*** |
|  | ***ses.setId("1015678");*** |
|  | ***Session prueba = facade.guardarSesion(ses);*** |
|  | ***assertEquals("El numero de sesion deberia ser el mismo",ses.getSession(),prueba.getSession());*** |
|  | ***assertEquals("El ID deberia ser el mismo",ses.getId(),prueba.getId());*** |
|  | ***}*** |
|  | ***}*** |

***Proxy Test unitario***

|  |  |
| --- | --- |
| ***package test;*** |  |
|  |  |
|  | ***import static org.junit.Assert.\*;*** |
|  |  |
|  | ***import org.junit.Test;*** |
|  |  |
|  | ***import com.example.echo.Proxy;*** |
|  | ***import com.example.echo.Session;*** |
|  |  |
|  | ***public class ProxyUTest {*** |
|  | ***Session ses = new Session();*** |
|  | ***Proxy proxy;*** |
|  |  |
|  | ***private void setupEscenario1() {*** |
|  | ***proxy = Proxy.rConstructora();*** |
|  | ***proxy.crearPasajero("mate.balles", "12345", "Julian", "Ballesteros", 21, "987654");*** |
|  | ***}*** |
|  | ***private void setupEscenario2() {*** |
|  | ***proxy = Proxy.rConstructora();*** |
|  | ***proxy.crearConductor("juancapoar", "54321", "Juan", "Posada", 21, "169701");*** |
|  | ***}*** |
|  |  |
|  |  |
|  | ***@Test*** |
|  | ***public void crearPasajeroTest() {*** |
|  | ***Proxy proxy = Proxy.rConstructora();*** |
|  | ***boolean creado = proxy.crearPasajero("mate.balles", "12345", "Julian", "Ballesteros", 21, "987654");*** |
|  | ***assertTrue("El usuario se debio crear",creado);*** |
|  | ***}*** |
|  | ***@Test*** |
|  | ***public void crearConductorTest() {*** |
|  | ***Proxy proxy = Proxy.rConstructora();*** |
|  | ***boolean creado =proxy.crearConductor("juancapoar", "54321", "Juan", "Posada", 21, "169701");*** |
|  | ***assertTrue("El usuario se debio crear",creado);*** |
|  | ***}*** |
|  |  |
|  | ***@Test*** |
|  | ***public void authTest1() {*** |
|  | ***setupEscenario1();*** |
|  | ***try {*** |
|  | ***ses = proxy.auth("mate", "12345");*** |
|  | ***} catch (Exception e) {*** |
|  | ***assertEquals("Error sesion no encontrada",e.getMessage());*** |
|  | ***}*** |
|  | ***}*** |
|  |  |
|  | ***@Test*** |
|  | ***public void authTest2() {*** |
|  | ***setupEscenario1();*** |
|  | ***try {*** |
|  | ***ses = proxy.auth("mate.balles", "12345");*** |
|  | ***assertEquals(ses.getId(),"987654");*** |
|  | ***} catch (Exception e) {*** |
|  | ***fail("Debio encontrar la sesion");*** |
|  | ***}*** |
|  |  |
|  | ***}*** |
|  |  |
|  | ***@Test*** |
|  | ***public void authTest3() {*** |
|  | ***setupEscenario2();*** |
|  | ***try {*** |
|  | ***ses = proxy.auth("juancapoar", "54321");*** |
|  | ***assertEquals(ses.getId(),"169701");*** |
|  | ***} catch (Exception e) {*** |
|  | ***fail("Debio encontrar la sesion");*** |
|  | ***}*** |
|  |  |
|  | ***}*** |
|  |  |
|  | ***}*** |

***Pruebas de sistema***

|  |  |
| --- | --- |
| ***package test;*** |  |
|  |  |
|  | ***import static org.junit.Assert.\*;*** |
|  |  |
|  | ***import java.util.ArrayList;*** |
|  |  |
|  | ***import org.junit.Test;*** |
|  |  |
|  | ***import com.example.echo.Facade;*** |
|  | ***import com.example.echo.IPago;*** |
|  | ***import com.example.echo.Pago;*** |
|  | ***import com.example.echo.Proxy;*** |
|  | ***import com.example.echo.Session;*** |
|  |  |
|  | ***public class PruebasSistema {*** |
|  | ***Proxy proxy;*** |
|  | ***Facade facade;*** |
|  | ***Session ses;*** |
|  | ***IPago pago = new Pago();*** |
|  | ***IPago epago;*** |
|  |  |
|  | ***public void setupEscenario1() {*** |
|  | ***proxy = Proxy.rConstructora();*** |
|  | ***facade = Facade.rConstructora();*** |
|  | ***proxy.crearPasajero("mate.balles", "12345", "Julian", "Ballesteros", 21, "987654");*** |
|  | ***try {*** |
|  | ***ses = proxy.auth("mate.balles", "12345");*** |
|  | ***} catch (Exception e) {*** |
|  | ***fail("Debio encontrar la sesion");*** |
|  | ***}*** |
|  | ***pago.setValores("3000,4567655," + ses.getId() + ",12345,Multa");*** |
|  | ***}*** |
|  |  |
|  | ***public void setupEscenario2() {*** |
|  | ***proxy = Proxy.rConstructora();*** |
|  | ***facade = Facade.rConstructora();*** |
|  | ***proxy.crearPasajero("mate.balles", "12345", "Julian", "Ballesteros", 21, "987654");*** |
|  | ***}*** |
|  |  |
|  | ***@Test*** |
|  | ***public void pagoSEfectivo() {*** |
|  | ***setupEscenario2();*** |
|  | ***try {*** |
|  | ***ses = proxy.auth("mate.balles", "12345");*** |
|  | ***} catch (Exception e) {*** |
|  | ***fail("Debio encontrar la sesion");*** |
|  | ***}*** |
|  | ***try {*** |
|  | ***IPago result;*** |
|  | ***pago.setValores("3000,4567655," + ses.getId() + ",12345,Viaje");*** |
|  | ***result = facade.pagoEfectivo(ses.getSession(), (Pago) pago, "11-19-18");*** |
|  | ***assertEquals("La referencia debe ser la misma", pago.getReferencia(), result.getReferencia());*** |
|  | ***} catch (Exception e) {*** |
|  | ***fail("Se debio realizar el pago efectivo");*** |
|  | ***}*** |
|  |  |
|  | ***}*** |
|  |  |
|  | ***@Test*** |
|  | ***public void pagoSEfectivo2() {*** |
|  | ***setupEscenario2();*** |
|  | ***try {*** |
|  | ***ses = proxy.auth("mate.balles", "12345");*** |
|  | ***} catch (Exception e) {*** |
|  | ***fail("Debio encontrar la sesion");*** |
|  | ***}*** |
|  | ***try {*** |
|  | ***IPago result;*** |
|  | ***pago.setValores("3000,4567655," + ses.getId() + ",12345,Viaje");*** |
|  | ***result = facade.pagoEfectivo(ses.getSession()+1 , (Pago) pago, "11-19-18");*** |
|  | ***} catch (Exception e) {*** |
|  | ***assertEquals("Error no se pudo realizar el pago en efectivo", e.getMessage());*** |
|  | ***}*** |
|  |  |
|  | ***}*** |
|  |  |
|  | ***@Test*** |
|  | ***public void pagoSDebito() {*** |
|  | ***setupEscenario2();*** |
|  | ***try {*** |
|  | ***ses = proxy.auth("mate.balles", "12345");*** |
|  | ***} catch (Exception e) {*** |
|  | ***fail("Debio encontrar la sesion");*** |
|  | ***}*** |
|  | ***try {*** |
|  | ***IPago result;*** |
|  | ***pago.setValores("3000,4567655," + ses.getId() + ",12345,Viaje");*** |
|  | ***result = facade.pagoDebito(ses.getSession(), (Pago) pago, "11-19-18,89327548");*** |
|  | ***assertEquals("El usuario que paga debe ser el mismo", pago.getUsuarioPaga(), result.getUsuarioPaga());*** |
|  | ***} catch (Exception e) {*** |
|  | ***fail("Se debio realizar el pago debito");*** |
|  | ***}*** |
|  |  |
|  | ***}*** |
|  |  |
|  | ***@Test*** |
|  | ***public void pagoSDebito2() {*** |
|  | ***setupEscenario2();*** |
|  | ***try {*** |
|  | ***ses = proxy.auth("mate.balles", "12345");*** |
|  | ***} catch (Exception e) {*** |
|  | ***fail("Debio encontrar la sesion");*** |
|  | ***}*** |
|  | ***try {*** |
|  | ***IPago result;*** |
|  | ***pago.setValores("3000,4567655," + ses.getId() + ",12345,Viaje");*** |
|  | ***result = facade.pagoDebito(ses.getSession() + 1, (Pago) pago, "11-19-18,89327548");*** |
|  | ***} catch (Exception e) {*** |
|  | ***assertEquals("Error no se pudo realizar el pago por debito", e.getMessage());*** |
|  | ***}*** |
|  |  |
|  | ***}*** |
|  |  |
|  | ***@Test*** |
|  | ***public void pagoSCredito() {*** |
|  | ***setupEscenario2();*** |
|  | ***try {*** |
|  | ***ses = proxy.auth("mate.balles", "12345");*** |
|  | ***} catch (Exception e) {*** |
|  | ***fail("Debio encontrar la sesion");*** |
|  | ***}*** |
|  | ***try {*** |
|  | ***IPago result;*** |
|  | ***pago.setValores("3000,4567655," + ses.getId() + ",12345,Viaje");*** |
|  | ***result = facade.pagoCredito(ses.getSession(), (Pago) pago, "11-19-18,456788834032,555,2");*** |
|  | ***assertEquals("El usuario pagado debe ser el mismo", pago.getUsuarioPagado(), result.getUsuarioPagado());*** |
|  | ***} catch (Exception e) {*** |
|  | ***fail("Se debio realizar el pago credito");*** |
|  | ***}*** |
|  |  |
|  | ***}*** |
|  |  |
|  | ***@Test*** |
|  | ***public void pagoSCredito2() {*** |
|  | ***setupEscenario2();*** |
|  | ***try {*** |
|  | ***ses = proxy.auth("mate.balles", "12345");*** |
|  | ***} catch (Exception e) {*** |
|  | ***fail("Debio encontrar la sesion");*** |
|  | ***}*** |
|  | ***try {*** |
|  | ***IPago result;*** |
|  | ***pago.setValores("3000,4567655," + ses.getId() + ",12345,Viaje");*** |
|  | ***result = facade.pagoCredito(ses.getSession() + 1, (Pago) pago, "11-19-18,456788834032,555,2");*** |
|  | ***} catch (Exception e) {*** |
|  | ***assertEquals("Error no se pudo realizar el pago por credito", e.getMessage());*** |
|  | ***}*** |
|  |  |
|  | ***}*** |
|  |  |
|  | ***@Test*** |
|  | ***public void listarPagosTest() {*** |
|  | ***setupEscenario1();*** |
|  | ***try {*** |
|  | ***pago = facade.pagoEfectivo(ses.getSession(), (Pago) pago, "12-11-19");*** |
|  | ***ArrayList<IPago> list = facade.listarPagos(ses.getSession());*** |
|  | ***assertEquals("La referencia de los pagos deberia ser la misma", pago.getReferencia(),*** |
|  | ***list.get(0).getReferencia());*** |
|  | ***} catch (Exception e) {*** |
|  | ***fail("Se debio realizar el pago");*** |
|  | ***}*** |
|  | ***}*** |
|  |  |
|  | ***}*** |

***Test Integracion Credito***

|  |  |
| --- | --- |
| ***package test;*** |  |
|  |  |
|  | ***import static org.junit.Assert.\*;*** |
|  |  |
|  | ***import org.junit.Test;*** |
|  |  |
|  | ***import com.example.echo.Credito;*** |
|  | ***import com.example.echo.Pago;*** |
|  |  |
|  | ***public class testICredito {*** |
|  |  |
|  | ***Pago pago = new Pago();*** |
|  | ***Credito pagoC;*** |
|  |  |
|  | ***public void setupEscenario1() {*** |
|  |  |
|  | ***pago.setValores("5000,987654321,123456,987654,Multa");*** |
|  |  |
|  | ***try {*** |
|  | ***pagoC = new Credito(pago);*** |
|  | ***pagoC.setValores("hoy,1234567898745632,753,6");*** |
|  | ***} catch (Exception e) {*** |
|  | ***fail("No deberia generar excepcion. ");*** |
|  | ***}*** |
|  | ***}*** |
|  |  |
|  | ***public void setupEscenario2() {*** |
|  |  |
|  | ***pago.setValores("3000,123456789,987654,123456,Viaje");*** |
|  |  |
|  | ***try {*** |
|  | ***pagoC = new Credito(pago);*** |
|  | ***pagoC.setValores("ayer,9876543212365478,681,12");*** |
|  | ***} catch (Exception e) {*** |
|  | ***fail("No deberia generar excepcion. ");*** |
|  | ***}*** |
|  | ***}*** |
|  |  |
|  | ***public void setupEscenario3() {*** |
|  |  |
|  | ***pago.setValores("10000,456789123,789456,987654,Devolucion");*** |
|  |  |
|  | ***try {*** |
|  | ***pagoC = new Credito(pago);*** |
|  | ***pagoC.setValores("mediodia,6547539512584560,431,1");*** |
|  | ***} catch (Exception e) {*** |
|  | ***fail("No deberia generar excepcion. ");*** |
|  | ***}*** |
|  | ***}*** |
|  |  |
|  | ***@Test*** |
|  | ***public void testFechaPago1() {*** |
|  | ***setupEscenario1();*** |
|  | ***assertEquals("La fecha del sistema debe coincidir con la ingresada.", "hoy", pagoC.getFecha());*** |
|  | ***}*** |
|  |  |
|  | ***@Test*** |
|  | ***public void testFechaPago2() {*** |
|  | ***setupEscenario2();*** |
|  | ***assertEquals("La fecha del sistema debe coincidir con la ingresada.", "ayer", pagoC.getFecha());*** |
|  | ***}*** |
|  |  |
|  | ***@Test*** |
|  | ***public void testFechaPago3() {*** |
|  | ***setupEscenario3();*** |
|  | ***assertEquals("La fecha del sistema debe coincidir con la ingresada.", "mediodia", pagoC.getFecha());*** |
|  | ***}*** |
|  |  |
|  | ***@Test*** |
|  | ***public void testTarjetaPago1() {*** |
|  | ***setupEscenario1();*** |
|  | ***assertEquals("El numero de Tarjeta de credito debe coincidir con la ingresada.", "1234567898745632",*** |
|  | ***pagoC.getNumTarjeta());*** |
|  | ***}*** |
|  |  |
|  | ***@Test*** |
|  | ***public void testTarjetaPago2() {*** |
|  | ***setupEscenario2();*** |
|  | ***assertEquals("El numero de Tarjeta de credito debe coincidir con la ingresada.", "9876543212365478",*** |
|  | ***pagoC.getNumTarjeta());*** |
|  | ***}*** |
|  |  |
|  | ***@Test*** |
|  | ***public void testTarjetaPago3() {*** |
|  | ***setupEscenario3();*** |
|  | ***assertEquals("El numero de Tarjeta de credito debe coincidir con la ingresada.", "6547539512584560",*** |
|  | ***pagoC.getNumTarjeta());*** |
|  | ***}*** |
|  |  |
|  | ***@Test*** |
|  | ***public void testSeguCodPago1() {*** |
|  | ***setupEscenario1();*** |
|  | ***assertEquals("El codigo de seguridad debe coincidir con la ingresada.", "753", pagoC.getSegucode());*** |
|  | ***}*** |
|  |  |
|  | ***@Test*** |
|  | ***public void testSeguCodPago2() {*** |
|  | ***setupEscenario2();*** |
|  | ***assertEquals("El codigo de seguridad debe coincidir con la ingresada.", "681", pagoC.getSegucode());*** |
|  | ***}*** |
|  |  |
|  | ***@Test*** |
|  | ***public void testSeguCodPago3() {*** |
|  | ***setupEscenario3();*** |
|  | ***assertEquals("El codigo de seguridad debe coincidir con la ingresada.", "431", pagoC.getSegucode());*** |
|  | ***}*** |
|  |  |
|  | ***@Test*** |
|  | ***public void testCuotasPago1() {*** |
|  | ***setupEscenario1();*** |
|  | ***assertEquals("La cantidad de cuotas debe coincidir con la ingresada.", "6", pagoC.getCuotas());*** |
|  | ***}*** |
|  |  |
|  | ***@Test*** |
|  | ***public void testCuotasPago2() {*** |
|  | ***setupEscenario2();*** |
|  | ***assertEquals("La cantidad de cuotas debe coincidir con la ingresada.", "12", pagoC.getCuotas());*** |
|  | ***}*** |
|  |  |
|  | ***@Test*** |
|  | ***public void testCuotasPago3() {*** |
|  | ***setupEscenario3();*** |
|  | ***assertEquals("La cantidad de cuotas debe coincidir con la ingresada.", "1", pagoC.getCuotas());*** |
|  | ***}*** |
|  |  |
|  | ***}*** |

***Test Integracion Debito***

|  |  |
| --- | --- |
| ***package test;*** |  |
|  |  |
|  | ***import static org.junit.Assert.\*;*** |
|  |  |
|  | ***import org.junit.Test;*** |
|  |  |
|  | ***import com.example.echo.Debito;*** |
|  | ***import com.example.echo.Pago;*** |
|  |  |
|  | ***public class testIDebito {*** |
|  |  |
|  | ***Pago pago = new Pago();*** |
|  | ***Debito pagoD;*** |
|  |  |
|  | ***public void setupEscenario1() {*** |
|  |  |
|  | ***pago.setValores("5000,987654321,123456,987654,Multa");*** |
|  |  |
|  | ***try {*** |
|  | ***pagoD = new Debito(pago);*** |
|  | ***pagoD.setValores("hoy,1234567898745632,753,6");*** |
|  | ***} catch (Exception e) {*** |
|  | ***fail("No deberia generar excepcion. ");*** |
|  | ***}*** |
|  | ***}*** |
|  |  |
|  | ***public void setupEscenario2() {*** |
|  |  |
|  | ***pago.setValores("3000,123456789,987654,123456,Viaje");*** |
|  |  |
|  | ***try {*** |
|  | ***pagoD = new Debito(pago);*** |
|  | ***pagoD.setValores("ayer,9876543212365478,681,12");*** |
|  | ***} catch (Exception e) {*** |
|  | ***fail("No deberia generar excepcion. ");*** |
|  | ***}*** |
|  | ***}*** |
|  |  |
|  | ***public void setupEscenario3() {*** |
|  |  |
|  | ***pago.setValores("10000,456789123,789456,987654,Devolucion");*** |
|  |  |
|  | ***try {*** |
|  | ***pagoD = new Debito(pago);*** |
|  | ***pagoD.setValores("mediodia,6547539512584560,431,1");*** |
|  | ***} catch (Exception e) {*** |
|  | ***fail("No deberia generar excepcion. ");*** |
|  | ***}*** |
|  | ***}*** |
|  |  |
|  | ***@Test*** |
|  | ***public void testFechaPago1() {*** |
|  | ***setupEscenario1();*** |
|  | ***assertEquals("La fecha del sistema debe coincidir con la ingresada.", "hoy", pagoD.getFecha());*** |
|  | ***}*** |
|  |  |
|  | ***@Test*** |
|  | ***public void testFechaPago2() {*** |
|  | ***setupEscenario2();*** |
|  | ***assertEquals("La fecha del sistema debe coincidir con la ingresada.", "ayer", pagoD.getFecha());*** |
|  | ***}*** |
|  |  |
|  | ***@Test*** |
|  | ***public void testFechaPago3() {*** |
|  | ***setupEscenario3();*** |
|  | ***assertEquals("La fecha del sistema debe coincidir con la ingresada.", "mediodia", pagoD.getFecha());*** |
|  | ***}*** |
|  |  |
|  | ***@Test*** |
|  | ***public void testCuentaPago1() {*** |
|  | ***setupEscenario1();*** |
|  | ***assertEquals("El numero de cuenta debe coincidir con la ingresada.", "1234567898745632", pagoD.getNumCuenta());*** |
|  | ***}*** |
|  |  |
|  | ***@Test*** |
|  | ***public void testCuentaPago2() {*** |
|  | ***setupEscenario2();*** |
|  | ***assertEquals("El numero de cuenta debe coincidir con la ingresada.", "9876543212365478", pagoD.getNumCuenta());*** |
|  | ***}*** |
|  |  |
|  | ***@Test*** |
|  | ***public void testCuentaPago3() {*** |
|  | ***setupEscenario3();*** |
|  | ***assertEquals("El numero de cuenta debe coincidir con la ingresada.", "6547539512584560", pagoD.getNumCuenta());*** |
|  | ***}*** |
|  |  |
|  | ***}*** |

***Test de integración Efectivo***

|  |  |
| --- | --- |
| package test; |  |
|  |  |
|  | import static org.junit.Assert.\*; |
|  |  |
|  | import org.junit.Test; |
|  |  |
|  | import com.example.echo.Efectivo; |
|  | import com.example.echo.Pago; |
|  |  |
|  | public class testIEfectivo { |
|  |  |
|  | Pago pago = new Pago(); |
|  | Efectivo pagoE; |
|  |  |
|  | public void setupEscenario1() { |
|  |  |
|  | pago.setValores("5000,987654321,123456,987654,Multa"); |
|  |  |
|  | try { |
|  | pagoE = new Efectivo(pago); |
|  | pagoE.setValores("hoy"); |
|  | } catch (Exception e) { |
|  | fail("No deberia generar excepcion. "); |
|  | } |
|  | } |
|  |  |
|  | public void setupEscenario2() { |
|  |  |
|  | pago.setValores("3000,123456789,987654,123456,Viaje"); |
|  |  |
|  | try { |
|  | pagoE = new Efectivo(pago); |
|  | pagoE.setValores("ayer"); |
|  | } catch (Exception e) { |
|  | fail("No deberia generar excepcion. "); |
|  | } |
|  | } |
|  |  |
|  | public void setupEscenario3() { |
|  |  |
|  | pago.setValores("10000,456789123,789456,987654,Devolucion"); |
|  |  |
|  | try { |
|  | pagoE = new Efectivo(pago); |
|  | pagoE.setValores("mediodia"); |
|  | } catch (Exception e) { |
|  | fail("No deberia generar excepcion. "); |
|  | } |
|  | } |
|  |  |
|  | @Test |
|  | public void testFechaPago1() { |
|  | setupEscenario1(); |
|  | assertEquals("La fecha del sistema debe coincidir con la ingresada.", "hoy", pagoE.getFecha()); |
|  | } |
|  |  |
|  | @Test |
|  | public void testFechaPago2() { |
|  | setupEscenario2(); |
|  | assertEquals("La fecha del sistema debe coincidir con la ingresada.", "ayer", pagoE.getFecha()); |
|  | } |
|  |  |
|  | @Test |
|  | public void testFechaPago3() { |
|  | setupEscenario3(); |
|  | assertEquals("La fecha del sistema debe coincidir con la ingresada.", "mediodia", pagoE.getFecha()); |
|  | } |
|  | } |

**Test Unitario Pago**

|  |  |
| --- | --- |
| **package test;** |  |
|  |  |
|  | **import static org.junit.Assert.\*;** |
|  |  |
|  | **import org.junit.Test;** |
|  |  |
|  | **import com.example.echo.Pago;** |
|  |  |
|  | **public class testUPago {** |
|  |  |
|  | **Pago pago = new Pago();** |
|  |  |
|  | **public void setupEscenario1() {** |
|  | **try {** |
|  | **pago.setValores("5000,987654321,123456,987654,Multa");** |
|  | **} catch (Exception e) {** |
|  | **fail("No deberia generar excepcion. ");** |
|  | **}** |
|  | **}** |
|  |  |
|  | **public void setupEscenario2() {** |
|  | **try {** |
|  | **pago.setValores("3000,123456789,987654,123456,Viaje");** |
|  | **} catch (Exception e) {** |
|  | **fail("No deberia generar excepcion. ");** |
|  | **}** |
|  | **}** |
|  |  |
|  | **public void setupEscenario3() {** |
|  | **try {** |
|  | **pago.setValores("10000,456789123,789456,987654,Devolucion");** |
|  | **} catch (Exception e) {** |
|  | **fail("No deberia generar excepcion. ");** |
|  | **}** |
|  | **}** |
|  |  |
|  | **@Test** |
|  | **public void testValorPago1() {** |
|  | **setupEscenario1();** |
|  | **assertEquals("El Valor del pago deberia ser igual.","5000", pago.getValor());** |
|  | **}** |
|  |  |
|  | **@Test** |
|  | **public void testValorPago2() {** |
|  | **setupEscenario2();** |
|  | **assertEquals("El Valor del pago deberia ser igual.","3000", pago.getValor());** |
|  | **}** |
|  |  |
|  | **@Test** |
|  | **public void testValorPago3() {** |
|  | **setupEscenario3();** |
|  | **assertEquals("El Valor del pago deberia ser igual.","10000", pago.getValor());** |
|  | **}** |
|  |  |
|  | **@Test** |
|  | **public void testRefPago1() {** |
|  | **setupEscenario1();** |
|  | **assertEquals("La referencia del pago deberia ser igual.","987654321", pago.getReferencia());** |
|  | **}** |
|  |  |
|  | **@Test** |
|  | **public void testRefPago2() {** |
|  | **setupEscenario2();** |
|  | **assertEquals("La referencia del pago deberia ser igual.","123456789", pago.getReferencia());** |
|  | **}** |
|  |  |
|  | **@Test** |
|  | **public void testRefPago3() {** |
|  | **setupEscenario3();** |
|  | **assertEquals("La referencia del pago deberia ser igual.","456789123", pago.getReferencia());** |
|  | **}** |
|  |  |
|  | **@Test** |
|  | **public void testUsuarioPaga1() {** |
|  | **setupEscenario1();** |
|  | **assertEquals("El id de Usuario que paga deberia ser igual.","123456", pago.getUsuarioPaga());** |
|  | **}** |
|  |  |
|  | **@Test** |
|  | **public void testUsuarioPaga2() {** |
|  | **setupEscenario2();** |
|  | **assertEquals("El id de Usuario que paga deberia ser igual.","987654", pago.getUsuarioPaga());** |
|  | **}** |
|  |  |
|  | **@Test** |
|  | **public void testUsuarioPaga3() {** |
|  | **setupEscenario3();** |
|  | **assertEquals("El id de Usuario que paga deberia ser igual.","789456", pago.getUsuarioPaga());** |
|  | **}** |
|  |  |
|  | **@Test** |
|  | **public void testUsuarioPagado1() {** |
|  | **setupEscenario1();** |
|  | **assertEquals("El id de Usuario que recibe el pago deberia ser igual.","987654", pago.getUsuarioPagado());** |
|  | **}** |
|  |  |
|  | **@Test** |
|  | **public void testUsuarioPagado2() {** |
|  | **setupEscenario2();** |
|  | **assertEquals("El id de Usuario que recibe el pago deberia ser igual.","123456", pago.getUsuarioPagado());** |
|  | **}** |
|  |  |
|  | **@Test** |
|  | **public void testUsuarioPagado3() {** |
|  | **setupEscenario3();** |
|  | **assertEquals("El id de Usuario que recibe el pago deberia ser igual.","987654", pago.getUsuarioPagado());** |
|  | **}** |
|  |  |
|  | **@Test** |
|  | **public void testConceptoPago1() {** |
|  | **setupEscenario1();** |
|  | **assertEquals("El concepto del pago deberia ser igual.","Multa", pago.getConcepto());** |
|  | **}** |
|  |  |
|  | **@Test** |
|  | **public void testConceptoPago2() {** |
|  | **setupEscenario2();** |
|  | **assertEquals("El concepto del pago deberia ser igual.","Viaje", pago.getConcepto());** |
|  | **}** |
|  |  |
|  | **@Test** |
|  | **public void testConceptoPago3() {** |
|  | **setupEscenario3();** |
|  | **assertEquals("El concepto del pago deberia ser igual.","Devolucion", pago.getConcepto());** |
|  | **}** |
|  | **}** |